ZYMON 2
MANUAL

Interak

DOCUMENT REF: UM-4/3
ISSUE: 3
DATED: JUNE 1982, (REPRINTED FEBRUARY 1983 AND MARCH 1984)

```
ZZZZZZ   Y   Y   M     M    OOO    N     N        222
    Z     Y Y    MM   MM   O   O   NN    N      2   2
   Z       Y     M MM M   O   O   N N   N          2
  Z        Y     M    M   O   O   N  N  N         22
 Z         Y     M    M   O   O   N   NN         2
ZZZZZZ     Y     M    M    OOO    N    N       22222
```

A MACHINE CODE MONITOR, DESIGNED AND BUILT BY BOB ELDRIDGE.

ZYMON 2 IS DEDICATED TO CHRISTINE.

THE AUTHOR WISHES TO THANK DAVID PARKINS OF GREENBANK ELECTRONICS FOR HIS INVALUABLE AID IN 1981 DURING THE PROTOTYPE BUILD OF ZYMON 1 (ON WHICH ZYMON 2 IS BASED).

CONTENTS

CONTROL SHEETS

INTRODUCTION

COMMANDS

IMPLEMENTATION

INFORMATION

NOTATION

UNLESS OTHERWISE STATED ALL NUMBERS ARE IN HEXADECIMAL (HEX.).

ABREVIATIONS AND DEFINITIONS

CR.....CARRIAGE RETURN, ALSO CAN BE CTRL-M.
EXT.....END OF TEXT, ALSO CAN BE CTRL-C.
FF.....FORM FEED, ALSO CAN BE CTRL-L.
BS.....BACKSPACE, ALSO CAN BE CTRL-H.

DD.....TWO HEX DIGITS, REPRESENTING AN 8 BIT BYTE.
HHHH.....FOUR HEX DIGITS, REPRESENTING A 16 BIT WORD.
ADDR.....FOUR HEX DIGITS, REPRESENTING A 16 BIT ADDRESS.

CS.....CHECKSUM = THE SUM OF A GROUP OF BYTES, CARRY IGNORED.

K.....1024 DECIMAL

PAGE.....4K BYTES THAT START ON A 4K BOUNDARY AS:-

PAGE 0 IS FROM 0000-0FFF.
PAGE 1 IS FROM 1000-1FFF.
PAGE E IS FROM E000-EFFF.

CHAPTER...64K BYTES THAT START ON A 64K BOUNDARY.

■.....REPRESENTS THE CURSOR, WHICH IS IN REVERSE VIDEO.

INTRODUCTION

ZYMON IS A MACHINE CODE UTILITY, DESIGNED TO RUN ON Z80-BASED MICRO-COMPUTERS, HAVING A MEMORY-MAPPED VDU. IT PROVIDES THE INTERFACE BETWEEN THE USER AND THE COMPUTER, ALLOWING MACHINE CODE PROGRAMS TO BE ENTERED, DEBUGGED AND RUN.

INSTALLATION CONCEPT

ZYMON IS INTENDED TO BE LOCATED AT PAGE E, E000-E7FF, IN THE MEMORY MAP, BUT IN MINIMUM SYSTEMS IT CAN BE LOCATED AT PAGE 0, 0000-07FF.

IF LOCATED AT PAGE E THEN A POWER ON JUMP TO E000 IS EXPECTED, AND RAM MUST OCCUPY PAGE 0, 0000-0FFF.

IF LOCATED AT PAGE 0, RAM MUST OCCUPY 0FC6-0FFF INCLUSIVE, SAY 2K AT 0800. (NOTE THAT AN MZB-CPU CARD NEEDS MODIFYING TO PERMIT RAM AND EPROM ON THE SAME PAGE.)

IN THE LATTER CASE PAGE E IS AVAILABLE FOR ROM OR RAM, WITH THE ONE EXCEPTION THAT IF LOCATION E000 IS ROM IT MUST NOT HOLD HEX 21. ALL OTHER CODES ARE ALLOWED.

THE GHOST REGISTER SET

THE Z80 REGISTERS PC, SP, AF, BC, DE, HL, IX AND IY ARE GHOSTED INTO RAM. THIS MEANS THAT ZYMON MAINTAINS AN AREA OF STORE (FROM 0FC6-OFD5) WITH A COPY OF THE ABOVE REGISTERS HELD IN IT. THESE GHOSTS ARE LOADED INTO THE Z80 WHENEVER AN EXECUTE COMMAND IS USED, SO BY PRELOADING THEM YOU CAN ENTER A PROGRAM AT ANY GIVEN ADDRESSS IN ORDER TO OBSERVE THE EFFECTS.

WHEN A SOFTWARE TRAP IS USED, DURING A DEBUG SESSION, THE GHOST REGISTERS RETAIN THE STATUS OF THE Z80. THIS ALLOWS YOU TO CONTINUE THE PROGRAM EXECUTION FROM THE TRAPPED ADDRESS AS IF THE TRAP HAD NOT TAKEN PLACE.

THE GHOST REGISTER SET MAY BE ALTERED USING THE MODIFY COMMAND, AND DISPLAYED USING THE REGISTERS COMMAND. THEY MAY BE SET TO ZERO USING THE ZERO COMMAND.

USER OPTION

THE INITIALISATION VALUES OF THE GHOST REGISTERS MAY BE CHANGED BY THE USER. TO DO THIS ALTER THE CONTENTS OF THE FOLLOWING ADDRESSES:-

ADDR	NOW	REGISTER	
E028	00	C	PC IS PROGRAM COUNTER
E029	00	P	(LOW BYTE FIRST)
E02E	C5	P	SP IS STACK POINTER
E02F	0F	S	(LOW BYTE FIRST)
E024	00	THE OTHERS.	

COMMAND FORMAT

COMMANDS GIVEN TO ZYMON FOLLOW A VERY RIGID SYNTAX STRUCTURE. THIS IS TO ENSURE THAT ZYMON ONLY DOES WHAT YOU WISH IT TO DO! (IF YOU HAVE EVER OVER-WRITTEN A 2K PROGRAM BY MISTYPING A COMMAND, YOU WILL APPRECIATE THIS POINT.)

THE EXPECTED FORMAT IS:-

COMMAND LETTER, ,PARAMETER 1, ,PARAMETER 2, ,PARAMETER 3

THE EMBEDDED SPACES (SHOWN AS ", ,") ARE MANDATORY. NOT EVERY COMMAND REQUIRES ALL THREE PARAMETERS. ERROR CODES ARE PRINTED IN THE EVENT OF MISTYPING ETC. THE CURSOR WILL ATTEMPT TO HIGHLIGHT ERRORS IN AN INCORRECT INPUT LINE.

SOME COMMANDS ARE CHAINED, THAT IS ZYMON WILL SET UP THE NEXT SEQUENTIAL COMMAND AFTER COMPLETING THE PREVIOUS ONE. TO EXIT FROM A CHAIN USE CTRL-C (EXT). COMMANDS WHICH ARE NOT CHAINED WILL TERMINATE WITH THE MESSAGE "ENTER COMMAND" WRITTEN TO LINE 23 ON THE VDU.

COLD START

AT SWITCH ON OR RESET ZYMON WILL INITIALISE THE SCRATCHPAD AND THE GHOST REGISTER SET, THEN IT WILL CLEAR THE VDU SCREEN. IF ZYMON IS AT PAGE E IT WILL MOVE ITSELF TO PAGE 0 AND BEGIN EXECUTION IN PAGE 0.

THE GHOST PC IS INITIALISED TO 0000, BUT WILL TAKE UP THE LAST USED VALUE ONCE THE E COMMAND HAS BEEN ENTERED.

WARM START

IF IT IS ARRANGED TO PULL THE NMI PIN 17 OF THE Z80 TO ZERO VOLTS MOMENTARILY, THE FACILITY OF WARM START CAN BE ADDED. THIS DIFFERS FROM COLD START IN THAT NO WORKSPACE OR GHOST REGISTERS ARE ALTERED, ALSO THE SCREEN IS NOT CLEARED. TO IMPLEMENT THIS FACILITY USE A MONOSTABLE TO PULSE PIN 17 OF THE Z80 TO 0 VOLTS. FOR SIMPLICITY A PUSH BUTTON AND 1k PULL UP RESISTOR CONNECTED BETWEEN PIN 17 AND GROUND WILL WORK BUT IT MAY BOUNCE A LITTLE!

THE WARM START CAN ALSO BE PRODUCED BY SOFTWARE MEANS, BY EXECUTING A JUMP TO 0066, THE WARM START ADDRESS IN ZYMON.

FIRST MESSAGE

POWERING ON OR PRESSING COLD START (Z80 RESET), WILL CAUSE ZYMON TO INITIALISE AND THE SCREEN WILL CLEAR, DISPLAYING:-

```
 ZYMON 2.VXXX
 ENTER COMMAND
>■
```

(NOTE:- THE WARM START DISPLAY WILL BE THE SAME EXCEPT THAT THE SCREEN WILL NOT BE CLEARED.)

YOU ARE NOW IN THE COMMAND ENTRY ROUTINE AND BY TYPING ON THE KEYBOARD YOU MAY ENTER AND RUN MACHINE CODE PROGRAMS.

BURN COMMAND

COMMAND LETTER.....B
PARAMETER 1.....4 DIGIT HEX ADDRESS, SOURCE START.
PARAMETER 2.....4 DIGIT HEX ADDRESS, SOURCE END.
PARAMETER 3.....4 DIGIT HEX ADDRESS, DESTINATION START. (EPROM).
CHAINED.....NO.
RESTRICTIONS.....SOURCE START MUST BE LESS THAN OR EQUAL TO THE SOURCE END.
THE DESTINATION BLOCK MUST NOT LIE WITHIN THE SOURCE BLOCK.

THE EPROM DESTINATION BLOCK STARTING FROM PARAMETER 3 WILL BE PROGRAMMED WITH THE DATA BLOCK INDICATED BY PARAMETER 1 TO PARAMETER 2 INCLUSIVE. AS LITTLE AS ONE BYTE CAN BE PROGRAMMED.

E.G. B E000 E000 C000 CR....ZYMON WILL PROGRAM THE EPROM ADDRESS C000 WITH THE DATA FROM E000.

IF THE DESTINATION BLOCK IS NOT ALL FF'S YOU WILL BE GIVEN THE OPTION TO CONTINUE THE BURN OR TO QUIT. AT THE COMPLETION OF THE BURN AUTOMATIC VERIFICATION OF THE TWO BLOCKS (SOURCE=DESTINATION) WILL BE DONE. IN THE EVENT OF A NON-COMPARE THE FAILING ADDRESSES AND DATA WILL BE SHOWN.

TO SAFE STOP A "BURN IN PROGRESS" USE COLD OR WARM START.

AS THERE IS NO RESTRICTION ON BLOCK SIZE THEN 2K, 4K AND 8K EPROMS CAN BE PROGRAMMED, GIVEN THAT THE PROGRAMMING REQUIRES A FIXED LEVEL ON A PROGRAM ENABLE PIN, CONTROLLED BY PORT F0 BIT 1 AND A PULSED LEVEL ON A BURN PIN, CONTROLLED BY PORT F0 BIT 2. FOR DETAILED INFORMATION ON EPROM PROGRAMMING HARDWARE SEE GREENBANK ELECTRONICS' DOCUMENT AN-C24, WHICH DETAILS THE REQUIREMENTS TO CONVERT THE EXISTING 2708 PROGRAMMING CARD TO BURN THE MODERN TYPES OF EPROM. (ZYMON'S ROUTINES HAVE BEEN BUILT SPECIFICALLY FOR THIS CARD, OR A COMPATIBLE REPLACEMENT.)

WARNING

ZYMON IS DESIGNED TO RUN AT 4MHZ, WITH NO WAIT STATES, AND THE BURN ROUTINES ARE TIMED AT THIS CLOCK. A 2MHZ CLOCK WOULD RESULT IN DAMAGING 100% OVERBURNING, SEE USER OPTION 2 BELOW.

USER OPTIONS

1. IF DESIRED THE PROGRAMMER HARDWARE MAY BE POWER SEQUENCED. THIS WOULD ALLOW DEAD SOCKET INSERTION/REMOVAL OF THE EPROM. TO IMPLEMENT THIS IN HARDWARE CAUSE BIT 0 OF PORT F0 TO ENABLE +5V TO THE PROGRAMMER; THE +5V IN TURN TO SWITCH ON THE +25V. (SEE THE CONTENTS FOR THE EPROM POWER SEQUENCE FLOWCHART.)

2. TO ALTER THE PROGRAMMING PULSE DURATION, CHANGE THE CONTENTS OF LOCATION 04E1 WHILST 'SCOPING THE PROGRAMMING PIN OF THE EMPTY EPROM SOCKET AND DOING DUMMY BURNS. WHEN YOU ARE CONTENT WITH THE PULSE DURATION, REBURN A ZYMON ROM PUTTING THE NEW DURATION TO ADDRESS E4E1. (IN MY SYSTEM I USE 32 AT E4E1 FOR 4MHZ, AND 19 FOR 2MHZ, NO WAIT STATES.)

EXECUTE COMMAND

COMMAND LETTER.....E
PARAMETER 1.....OPTIONAL 4 DIGIT HEX ADDRESS, EXECUTE START.
PARAMETER 2.....OPTIONAL 4 DIGIT HEX ADDRESS, TRAP POINT.
PARAMETER 3.....NOT USED
CHAINED.....YES, IF A TRAP IS SET.
RESTRICTIONS.....TRAPS MUST BE SET IN RAM.

IF NO PARAMETERS ARE GIVEN ZYMON WILL SET UP AN EXECUTE COMMAND USING THE CONTENTS OF THE GHOST PC (G.PC) FOR PARAMETER 1.
IF PARAMETER 1 IS GIVEN ZYMON WILL LOAD THE Z80 FROM THE GHOST REGISTER SET AND EXECUTE CODE FROM THE ADDRESS GIVEN AS PARAMETER 1.
IF PARAMETER 1 AND PARAMETER 2 ARE GIVEN ZYMON WILL SET A TRAP AT THE ADDRESS GIVEN AS PARAMETER 2, SUCH THAT IF THE Z80 GETS TO IT, IT WILL DISPLAY THE Z80 STATUS ON THE VDU. THEN ZYMON WILL LOAD THE Z80 WITH THE GHOST REGISTER SET AND EXECUTE CODE FROM THE ADDRESS GIVEN AS PARAMETER 1.

IE. E CR.....WILL PRODUCE E G.PC, YOU MAY NOW PRESS CR OR SET A TRAP AND PRESS CR.

E 3000 CR.....WILL EXECUTE THE CODE STARTING AT 3000.

E 3000 300A CR.WILL SET A TRAP AT 300A AND EXECUTE CODE FROM ADDRESS 3000.

TRAPS

IF YOU COMMAND A TRAP AND PRESS CR ZYMON WILL FIRST SAVE THE ORIGINAL CONTENTS OF THAT LOCATION AND WRITE INTO IT HEX CODE FF. (RST 56 IN Z80 ASSEMBLY LANGUAGE).
THEN CODE WILL BE EXECUTED FROM PARAMETER 1.
WHEN THE Z80 GETS TO THE TRAP IT WILL STACK THE CURRENT PC CONTENTS AND PASS CONTROL TO ZYMON.
ZYMON WILL UNSTACK THE PC CONTENTS AND SAVE THIS AND THE OTHER Z80 STATUS IN THE GHOST REGISTER STORE. THEN ZYMON WILL DISPLAY THIS STATUS TO THE VDU SCREEN. THE ORIGINAL OP-CODE CONTENTS OF THE TRAPPED LOCATION ARE THEN RESTORED. FINALLY ZYMON WILL SET UP AN EXECUTE COMMAND USING THE TRAPPED ADDRESS AS THE EXECUTE PARAMETER. AT THIS POINT YOU MAY ENTER ANOTHER TRAP ADDRESS, PRESS CR AND STEP THE PROCESS ON TO OBSERVE THE STATUS AT THE NEXT TRAP POINT.
AS CAN BE REALISED THIS MECHANISM IS A POWERFUL DEBUGGING TOOL, ALLOWING A PROGRAM TO BE SYSTEMATICALLY STEPPED THROUGH TO EXAMINE ITS ACTIONS AT KEY PLACES IN THE LOGICAL FLOW.

NOTE:- YOU CAN OF COURSE SET ARTIFICIAL TRAPS BY WRITING HEX FF TO CHOSEN LOCATIONS AND ZYMON WILL OBLIGE BY DUMPING THE Z80 STATUS AT EACH ONE. BUT! AS ZYMON HAS NOT BEEN TOLD OF THE FUTURE TRAP IT WILL CONSIDER YOU TO HAVE MADE AN ERROR AND DISPLAY AN APPROPRIATE ERROR MESSAGE WITH THE STATUS.

AS AN ASIDE IT'S NOT A BAD IDEA TO FILL THE ENTIRE STORE WITH HEX FF BEFORE ENTERING THE PROGRAM UNDER TEST, AS IT'S THEN VERY HARD TO LOOSE CONTROL, AS AN UNEXPECTED TRAP IS ALMOST BOUND TO BE THE RESULT OF A PROGRAM CRASH.
(FF WAS CHOSEN AS NO STORE READS BACK AS FF, SO YOUR PROGRAM CANNOT RUN OFF THE END OF THE MEMORY SPACE!!).

COPY COMMAND

```
COMMAND LETTER.....C
PARAMETER 1........4 DIGIT HEX ADDRESS, SOURCE START.
PARAMETER 2........4 DIGIT HEX ADDRESS, SOURCE END.
PARAMETER 3........4 DIGIT HEX ADDRESS, DESTINATION START.
CHAINED............NO
RESTRICTIONS.......SOURCE START MUST BE LESS THAN OR EQUAL
                   TO SOURCE END.
```

A COPY OF THE CONTENTS OF PARAMETER 1 - PARAMETER 2 INCLUSIVE IS WRITTEN TO A DESTINATION BLOCK, STARTING FROM PARAMETER 3. THE COPY IS INTELLIGENT, THAT IS, PARAMETER 3 MAY BE WITHIN THE BLOCK DEFINED BY PARAMETER 1 - PARAMETER 2.

```
EG.  C 3000 33FF 3001 CR...THE 1K BLOCK FROM 3000-33FF WILL
                           BE MOVED ALONG BY ONE BYTE.
```

FILL COMMAND

```
COMMAND LETTER.....F
PARAMETER 1........4 DIGIT HEX ADDRESS, BLOCK START.
PARAMETER 2........4 DIGIT HEX ADDRESS, BLOCK END.
PARAMETER 3........2 DIGIT HEX VALUE, DATA TO BE WRITTEN.
CHAINED............NO
RESTRICTIONS.......BLOCK START MUST BE LESS THAN OR EQUAL
                   TO BLOCK END.
```

PARAMETER 3 WILL BE WRITTEN TO THE ENTIRE BLOCK AS DEFINED BY PARAMETER 1-PARAMETER 2 INCLUSIVE.

```
EG.  F 3000 33FF FF CR....THE 1K BLOCK FROM 3000-33FF WILL
                          HAVE DATA BYTE FF WRITTEN TO ALL
                          LOCATIONS.
```

JUMP COMMAND

```
COMMAND LETTER.....J
PARAMETER 1........4 DIGIT HEX ADDRESS, JUMP INSTRUCTION.
PARAMETER 2........4 DIGIT HEX ADDRESS, JUMP DESTINATION.
PARAMETER 3........NOT USED.
CHAINED............NO
RESTRICTIONS.......NONE
```

ZYMON WILL RETURN THE REQUIRED JUMP DISPLACEMENT.

```
EG.  J 300A 2FEC CR...WILL PRODUCE J 300A 2FEC F0.
                      WHERE E0 IS THE REQUIRED JUMP DISPLACEMENT
                      FOR A JUMP INSTRUCTION AT 300A JUMPING
                      TO ADDRESS 2FEC.
```

NOTE:- IF THE COMPUTED JUMP IS OUT OF THE RELATIVE RANGE AN ERROR J MESSAGE WILL BE OUTPUT. THE CURSOR WILL POSITION ITSELF INBETWEEN THE TWO ADDRESSES AS ZYMON CANNOT TELL WHICH IS THE INCORRECT VALUE.

MODIFY COMMAND

COMMAND LETTER.....M
PARAMETER 1.....4 DIGIT HEX ADDRESS, LOCATION.
PARAMETER 2.....OPTIONAL 2 DIGIT HEX BYTE, DATA.
PARAMETER 3.....NOT USED
CHAINED.....YES
RESTRICTIONS.....NONE.

IF PARAMETER 2 IS NOT SPECIFIED A READ OF THE ADDRESS GIVEN IN PARAMETER 1 WILL TAKE PLACE.
IF PARAMETER 2 IS SPECIFIED THE DATA GIVEN AS PARAMETER 2 WILL BE WRITTEN TO THE ADDRESS GIVEN AS PARAMETER 1. FOLLOWING THIS THE ADDRESS GIVEN AS PARAMETER 1 WILL BE READ BACK TO LINE 23 SO YOU CAN CONFIRM THE CONTENTS.

EG. M 3000 CR....ZYMON WILL RETURN THE CONTENTS. M 3000 DD.

M 3000 FF CR.FF IS WRITTEN TO ADDRESS 3000. THEN IT IS READ BACK TO LINE 23.

IT IS PERMITTED TO MODIFY ROM WITHOUT AN ERROR MESSAGE AS:-

1) THE AUTOMATIC READ BACK WILL INDICATE THAT THE DATA HAS NOT BEEN WRITTEN.

2) TO ALLOW EXTRA CONTROL PORTS TO BE REACHED BY WRITING TO ROM SPACE. (FOR INSTANCE THIS METHOD COULD BE USED TO SELECT EXTRA 64K MEMORY CHAPTERS IN VIRTUAL MACHINE SYSTEMS).

PORT COMMAND

COMMAND LETTER.....P
PARAMETER 1.....2 DIGIT HEX I/O ADDRESS, PORT.
PARAMETER 2.....OPTIONAL 2 DIGIT HEX BYTE, WRITE DATA.
PARAMETER 3.....NOT USED.
CHAINED.....NO
RESTRICTIONS.....NONE

IF PARAMETER 2 IS NOT SPECIFIED A READ OF THE PORT WILL TAKE PLACE.
IF PARAMETER 1 IS SPECIFIED THE DATA GIVEN AS PARAMETER 2 WILL BE WRITTEN TO THE PORT GIVEN AS PARAMETER 1. (AUTO READ BACK IS NOT IMPLEMENTED AS THE PORTS ARE OFTEN USED FOR CONTROL PURPOSES, AND ANYWAY THEY ARE OFTEN READ ONLY OR WRITE ONLY).

EG. P F0 CR.....ZYMON WILL RETURN THE PORT CONTENTS. P F0 FF

P F0 00 CR..00 IS WRITTEN TO PORT F0. (NO READ BACK).

USER OPTION

AUTOMATIC READ BACK OF THE PORT CAN BE IMPLEMENTED IF DESIRED. TO DO THIS ALTER ADDRESS 02C1 FROM DD TO F1, AND READ AFTER WRITE WILL OCCUR.

REGISTERS COMMAND

COMMAND LETTER.....R
PARAMETER 1.....NOT USED
PARAMETER 2.....NOT USED
PARAMETER 3.....NOT USED
CHAINED.....NO
RESTRICTIONS.....NONE

ZYMON WILL DISPLAY THE ENTIRE GHOST REGISTER SET.

EG. R CR...DISPLAYS THE GHOST REGISTER SET.

ZERO COMMAND

COMMAND LETTER.....Z
PARAMETER 1.....NOT USED
PARAMETER 2.....NOT USED
PARAMETER 3.....NOT USED
CHAINED.....NO
RESTRICTIONS.....NONE

ZYMON WILL ZERO THE GHOST REGISTER SET.

EG. Z CR..ZEROS AND DISPLAYS THE GHOST REGISTER SET.

TABULATE COMMAND

COMMAND LETTER.....T
PARAMETER 1.....4 DIGIT HEX ADDRESS, START POINT
PARAMETER 2.....NOT USED
PARAMETER 3.....NOT USED
CHAINED.....YES
RESTRICTIONS.....NONE

A TABULATION OF STORE TO SCREEN WILL BE PERFORMED.

EG. T 3000 CR...THE STORE FROM ADDRESS 3000 ONWARDS WILL BE
TABULATED ON THE VDU. 64 DECIMAL LOCATIONS.

USER OPTIONS

THE NUMBER OF LINES TO BE TABULATED CAN BE SET BY THE USER.
ALTER THE CONTENTS OF ADDRESS 024F TO THE DESIRED NUMBER.

THE NUMBER OF BYTES PER LINE CAN BE SET BY THE USER.
ALTER THE CONTENTS OF ADDRESS 0256 TO THE DESIRED NUMBER.

SAVE COMMAND

```
COMMAND LETTER.....S
PARAMETER 1........4 DIGIT HEX ADDRESS, SOURCE START.
PARAMETER 2........4 DIGIT HEX ADDRESS, SOURCE END.
PARAMETER 3........NOT USED.
CHAINED............NO
RESTRICTIONS.......SOURCE START MUST BE LESS THAN OR EQUAL
                   TO SOURCE END.
```

THE DATA FROM PARAMETER 1-PARAMETER 2 INCLUSIVE WILL BE SAVED TO THE CASSETTE RECORDER.

```
EG.  S 3000 33FF...DO NOT PRESS CR
                   PRESS RECORD AND PLAY ON THE TAPE.
                   PRESS CR.
```

ZYMON WILL SAVE THE 1K BLOCK OF DATA FROM 3000-33FF TO TAPE. WITH THE FOLLOWING FORMAT:-

```
   64 BYTES  00  LEADER
             FF  TAPE MARK
             BB
             CC  16 BIT BYTE COUNT
             SS
             AA  16 BIT START ADDRESS
             CS  8 BIT CHECKSUM. (BB+CC+SS+AA).
    8 BYTES  DD  8 DATA BYTES
             CS  8 BIT CHECKSUM. (DD+DD+DD+DD+DD+DD+DD+DD)
             --
             --
             --
    N BYTES  DD  LAST BLOCK OF N BYTES. (8 OR LESS)
             CS  LAST CHECKSUM. (DD+NDD"S).
```

DURING THE SAVE PROCESS THE DATA IS REFLECTED TO THE VDU (24) AT THE COMPLETION OF THE SAVE "ENTER COMMAND" IS DISPLAYED TO THE VDU.

- SEE PAGE 32b FOR USER CUSTOMISATION OF SAVE COMMAND -

VERIFY COMMAND

```
COMMAND LETTER.....V
PARAMETER 1........4 DIGIT HEX ADDRESS, SOURCE START
PARAMETER 2........4 DIGIT HEX ADDRESS, SOURCE END.
PARAMETER 3........4 DIGIT HEX ADDRESS, DESTINATION START
CHAINED............NO
RESTRICTIONS.......SOURCE START MUST BE LESS THAN OR EQUAL
                   TO SOURCE END.
```

A LOGICAL COMPARE OF THE TWO STORE BLOCKS WILL TAKE PLACE. IN THE EVENT OF A NON-COMPARE THE ERRORED ADDRESSES WILL BE DISPLAYED TO THE VDU.

```
EG.  V 3000 33FF 4000 CR...THE 1K BLOCK FROM 3000-33FF WILL
                           BE COMPARED WITH THE 1K BLOCK
                           THAT STARTS AT 4000.
```

LOAD COMMAND

COMMAND LETTER.....L
PARAMETER 1.....OPTIONAL 4 DIGIT HEX ADDRESS, LOAD POINT.
PARAMETER 2.....NOT USED
PARAMETER 3.....NOT USED
CHAINED.....NO
RESTRICTIONS.....NONE

IF PARAMETER 1 IS NOT SPECIFIED THE TAPE DATA WILL BE LOADED TO THE ADDRESS HELD ON THE TAPE.
IF PARAMETER 1 IS SPECIFIED THE TAPE DATA WILL BE LOADED TO STORE STARTING AT THE ADDRESS SPECIFIED BY PARAMETER 1.

EG. L CR.....THE TAPE DATA WILL BE LOADED TO STORE STARTING AT THE ADDRESS HELD ON THE TAPE.

L C000 CR.THE TAPE DATA WILL BE LOADED TO STORE STARTING AT THE ADDRESS SPECIFIED BY PARAMETER 1.

AFTER PRESSING CR ZYMON WILL ENTER A SEARCH LOOP EXPECTING TO FIND 32 (DEC) 00 BYTES FROM A TAPE LEADER. WHEN THESE HAVE BEEN DETECTED THE DATA THAT FOLLOWS WILL BE READ INTO STORE. THE VDU WILL DISPLAY THE DATA AS IT IS READ IN, AS:-

ADDR DD DD DD DD DD DD DD DD CS.

THE CS MEANS CHECKSUM AND IF THIS DOES NOT COMPARE WITH A COMPUTED CHECKSUM THE LINE IS SCROLLED UP THE SCREEN. EVEN SO THE DATA IS STILL WRITTEN TO STORE, SO THAT AT THE END OF THE LOAD ANY ERRORED LINES WILL BE VISIBLE ON THE SCREEN WITH THE BLOCK ADDRESS TO POINT TO ITS STORE POSITION. THIS ENABLES YOU TO CORRECT THE ONE OR TWO ERRORED BYTES WITHOUT HAVING TO COMPLETELY RELOAD THE FILE.
THERE IS ONE CASE THOUGH WHEN THE LOAD WILL BE TERMINATED WHEN AN ERROR IS FOUND, AND THIS IS WHEN THE ADDRESS OR TOTAL BYTE COUNT READ FROM THE TAPE, IS WRONG. IF THE LOAD WAS TO CONTINUE DATA COULD BE WRITTEN TO THE WRONG STORE ADDRESS WITH CATASTROPHIC RESULTS.
IN THIS CASE ZYMON WILL PRODUCE AN ERROR C MESSAGE AND DISPLAY TO LINE 24 THE OVERALL BYTE COUNT, THE TAPE READ ADDRESS AND THE RECEIVED CHECKSUM OF THEM BOTH. (BB+CC+SS+AA).
IE.

ERROR (C)
BBCC SSAA CS

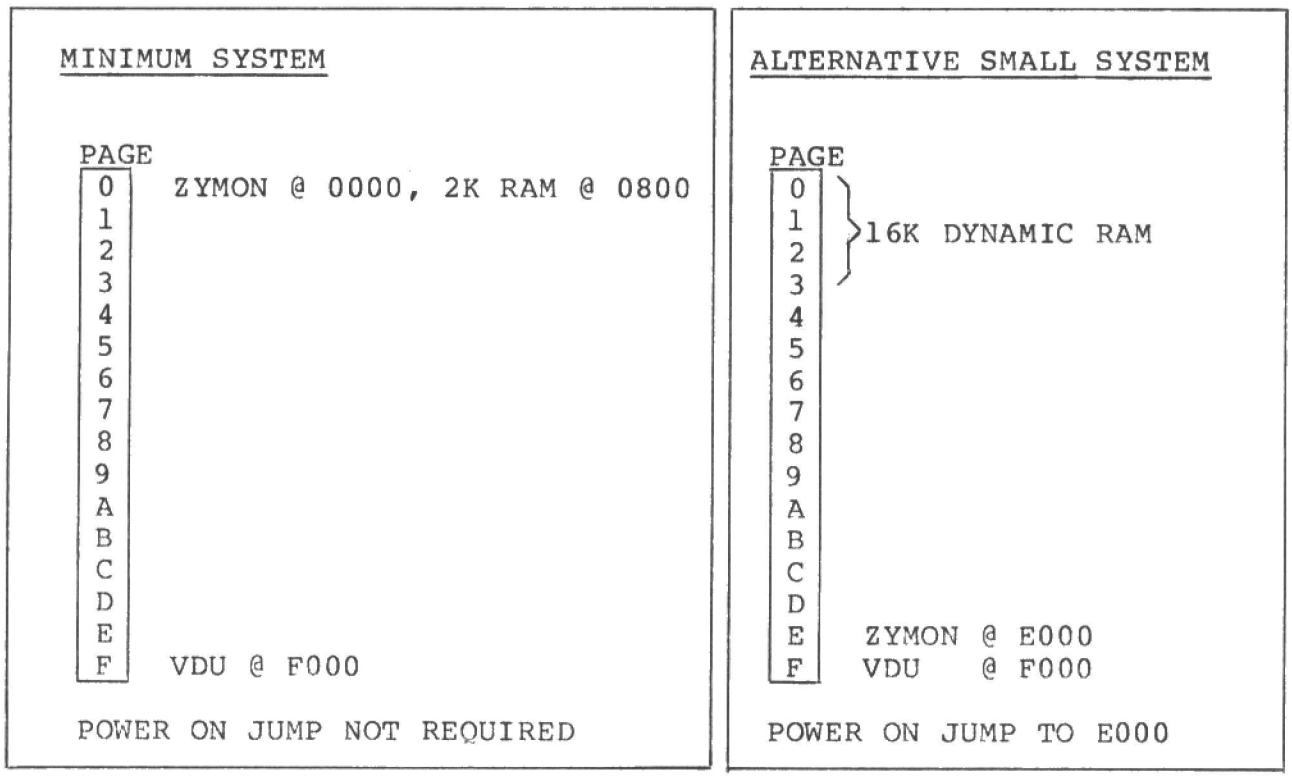
BECAUSE OF THE LEADER SEARCH YOU CAN AFTER PRESSING CR REWIND AND POSITION THE TAPE. ALSO YOU DON'T NEED TO BE WITHIN THE LEADER BEFORE LOADING, SO LONG AS YOU ARE BEFORE THE FILE ZYMON WILL WAIT FOR IT.

TO TERMINATE A PARTLY DONE LOAD USE COLD OR WARM START.

AT THE END OF THE LOAD ZYMON WILL INDICATE THE LOADED DATA BY DISPLAYING TO LINE 23 THE MESSAGE:-

L STARTADDRESS ENDADDRESS

MEMORY MAPS WITH ZYMON



EXPANDED INTERAK SYSTEM.

THIS WOULD BE BASED ON THE SYSTEM SHOWN ABOVE RIGHT, AND WOULD HAVE AT LEAST 48K OF RAM. PAGES C AND D ARE NOT RIGIDLY ALLOCATED, AND MOST CPU CARDS WITH POWER ON JUMP CIRCUITRY ALSO PERMIT THE ZYMON EPROM AT E000 TO BE SWITCHED OUT UNDER SOFTWARE CONTROL, LEAVING PAGE E AVAILABLE TO THE USER. IN SOME CIRCUMSTANCES (E.G. CP/M DISK SYSTEMS) THIS 12K COULD BE RAM WHICH COULD BE ADDED TO THE FIRST 48K MAKING 60K OF RAM IN ALL. MORE OFTEN THE C AND D SPACE IS USED FOR FIRMWARE AND/OR AN EPROM PROGRAMMER; TYPICALLY PAGE C FOR THE SUBJECT EPROM, PAGE D FOR THE MASTER EPROM, WITH REMAINING PAGE E SPACE IN RESERVE FOR ANY ADDITIONAL FIRMWARE NEEDS (E.G. FOR FLOPPY DISK BOOT ROM). SOME USERS SWITCH IN DIFFERENT THINGS AT PAGES C, D AND E, E.G. PROGRAMMER, EPROM FIRMWARE, OR RAM.

PORT UTILISATION SUMMARY

PORT NO.	USE
04	CASSETTE UART STATUS
05	CASSETTE UART DATA READ/WRITE
40	ASCII KEYBOARD, 7 BITS + STROBE
FO	EPROM PROGRAMMER CONTROL PORT

(THE NEXT PAGE GIVES MORE DETAILS OF THE USE OF THE BITS IN EACH PORT.)

PERIPHERAL MAPPING

VDU

24 LINES OF 32 CHARACTERS. THE VDU IS MEMORY MAPPED, THE TOP LEFT-HAND LOCATION BEING F000, CONTINUING IN SEQUENCE TO F2FF, THE BOTTOM RIGHT-HAND LOCATION. ZYMON DOES NOT USE THE LOWER CASE LETTERS, BUT THEIR PRESENCE PROVIDES NO DIFFICULTY.

CURSOR

THIS IS THE CHARACTER DISPLAYED IN REVERSE VIDEO. FOR THE CURSOR DISPLAY TO WORK AS INTENDED, THE VDU SHOULD HAVE THE OPTION SELECTED WHERE THE HIGH BIT (BIT 7) OF THE CHARACTER TO BE DISPLAYED CAUSES REVERSE VIDEO WHEN IT IS HIGH.

CASSETTE

UART DRIVEN 300-2400 BAUD.

PORT 04 IS THE STATUS PORT:-
STATUS BIT 7 IS TBMT (TRANSMIT BUFFER EMPTY)
STATUS BIT 6 IS DAV (DATA AVAILABLE)

PORT 05 IS THE DATA PORT:-
WHEN THE STATUS IS CORRECT THE DATA IS WRITTEN TO, OR READ FROM, PORT 05.

KEYBOARD

PORT 40 IS THE KEYBOARD PORT:-
ASCII POSITIVE LOGIC. CHARACTERS A TO Z MUST BE UPPER CASE ONLY (LOWER CASE CHARACTERS ARE NOT RECOGNISED). 7-BIT ASCII DATA ON THE LOWER 7 DATA LINES (BITS 0 TO 6), THE HIGH BIT (BIT 7) TO BE A POSITIVE GOING STROBE PULSE INDICATING THAT THE ASCII DATA IS VALID.

EPROM PROGRAMMER

2516 OR 5V-2716 EPROMS MAY BE PROGRAMMED (ALSO 4K AND 8K TYPES IF FUNDS PERMIT). THE MASTER (SOURCE) EPROM MAY BE MAPPED TO ANY FREE ADDRESS, LIKEWISE THE SUBJECT (DESTINATION) EPROM.

PORT F0 IS THE CONTROL PORT:-
CONTROL BIT 0 TURNS THE EPROM POWER SUPPLY ON (HARDWARE PERMITTING).
CONTROL BIT 1 TURNS ON THE +25V PROGRAMMING VOLTAGE.
CONTROL BIT 2 IS THE 50 ms PROGRAMMING PULSE.

MORE INFORMATION ON THIS SUBJECT CAN BE FOUND IN THE GREENBANK ELECTRONICS DOCUMENT AN-C24, WHICH DETAILS THE REQUIREMENTS FOR CONVERTING EXISTING 2708 EPROM PROGRAMMER CARDS TO THE MODERN TYPES. THE POWER SUPPLY SEQUENCING (CONTROL BIT 0) IS NOT IMPLEMENTED IN THE SIMPLE CONVERSION, BUT IF IT WAS IT WOULD GIVE THE BENEFIT OF BEING ABLE TO LOAD/UNLOAD THE EPROM FROM A DEAD FRONT PANEL SOCKET. SEE PAGE 36 FOR "EPROM POWER SEQUENCE FLOWCHART"

ZYMON 2


```
0000 2100E0    LD HL,E000      (COLD START)THE KEY IS E000,PUT IT IN HL.
0003 46        LD B,(HL)       SAVE KEY DATA.
0004 3600      LD (HL),00      ZERO THE KEY LOCATION.
0006 7E        LD A,(HL)       READ THE KEY LOCATION.
0007 70        LD (HL),B       RESTORE OLD KEY DATA.
0008 FE21      CP21            TEST KEY EQUAL TO 21 ?
000A 2010      JRNZ 10         NOT 21 SO JUMP TO (ZERO)
000C C30FE0    JP E00F         BRANCH TO E00F
000F DBFF      IN A,(FF)       SWITCH OFF THE POWER ON JUMP
0011 110000    LD DE,0000      POINT TO ZYMON'S HOME
0014 010008    LD BC,0800      LOAD ZYMON'S SIZE=2K BYTES
0017 EDB0      LDIR            TRANSFER ZYMON TO PAGE ZERO
0019 C31C00    JP 001C         BRANCH TO (ZERO)
001C 210010    LD HL,1000      (ZERO)POINT TO THE TOP OF ZYMONS RAM SPACE
001F F9        LD SP,HL        SET STACK TOP TO 1000,FIRST USED IS 0FFF.
0020 0636      LD B,36         GET THE SCRATCHPAD SIZE.
0022 2B        DEC HL          (CLEAR)POINT TO FIRST/NEXT LOCATION.
0023 3600      LD (HL),00      ZERO THE LOCATION
0025 10FB      DBJNZ,FB        IF NOT DONE JUMP TO (CLEAR)
0027 210000    LD HL,0000      SET THE COLD START AUTO-JUMP ADDRESS.
002A 22C60F    LD (0FC6),HL    STORE AUTO-JUMP IN THE GHOST PC.
002D 21C50F    LD HL,0FC5      GHOST STACK INITIALISATION VALUE.
0030 22C80F    LD (0FC8),HL    INITIALISE THE GHOST SP.
0033 CDC306    CALL 06C3       CLEAR THE VDU SCREEN.
0036 182E      JR 2E           JUMP TO (WARM START SKIP)
0038 22D40F    LD (0FD4),HL    (TRAP ENTRY)SAVE THE USER HL
003B E1        POP HL          GET THE USER'S PC
003C 2B        DEC HL          ADJUST IT BY -1.
003D 22C60F    LD (0FC6),HL    SAVE USER'S PC
0040 ED73C80F  LD (0FC8),SP    SAVE USER'S SP
0044 31D40F    LD SP,0FD4      POINT TO GHOST STORE.
0047 D5        PUSH DE         SAVE USER'S DE
0048 C5        PUSH BC         SAVE USER'S BC
0049 F5        PUSH AF         SAVE USER'S AF
004A FDE5      PUSH IY         SAVE USER'S IY
004C DDE5      PUSH IX         SAVE USER'S IX
004E 310010    LD SP,1000      POINT TO REAL STACK TOP.
0051 2AE40F    LD HL,(0FE4)    GET EXPECTED TRAP ADDRESS.
0054 E5        PUSH HL         STACK EXPECTED TRAP ADDRESS
0055 CD3506    CALL 0635       RESTORE USERS OPCODE VALUE
0058 CDB205    CALL 05B2       PRINT THE Z80 STATUS.
005B D1        POP DE          FETCH EXPECTED TRAP ADDRESS
000C 2AC60F    LD HL,(0FC6)    FETCH THE ACTUAL TRAP ADDRESS
005F CDBB06    CALL 06BB       COMPARE EXPECTED/ACTUAL FOR EQUALITY?
0062 3E55      LD A,55         ERROR U CODE,JUST IN CASE IT'S NEEDED.
0064 1802      JR 02           HOP OVER THE WARM START ENTRY POINT.
0066 180F      JR 0F           SKIP TO (REAL WARM START)
0068 CA4202    JPZ 0242        IF COMPARE TRUE JUMP TO (WAS 0)
006B 21E0F2    LD HL,F2E0      (ERROR)POINT TO LINE 24 START
006E CD0207    CALL 0702       CLEAR LINES 24 AND 23.
0071 363E      LD (HL),3E      PRINT THE PROMPT.
0073 23        INC HL          STEP POINTER ON BY ONE.
0074 C3FD00    JP 00FD         BRANCH TO (ERROR 1)
```

ØØF7

Ø242

ØØFD

```
0077 AF        XOR A          (REAL WARM START)IS HERE,ZERO A
0078 D3F0      OUT A,(F0)     SWITCH OFF THE PROGRAMMER
007A 310010    LD SP,1000     INITIALISE THE STACK
007D 216B07    LD HL,076B     POINT TO STRING "ZYMON 2"
0080 CDEF05    CALL 05EF      PRINT "ZYMON 2"
0083 CD0206    CALL 0602      SCROLL THE SCREEN
0086 CD3506    CALL 0635      RESTORE USERS OP-CODE
0089 000000    NOP,NOP,NOP    CONTINUE TO 008C
008C 1803      JR 03          JUMP TO (ENTCOM)
008E CDC306    CALL 06C3      (CVDU)CLEAR THE VDU
0091 21E0F2    LD HL,F2E0     (ENTCOM)POINT TO LINE 24 START
0094 CD0207    CALL 0702      CLEAR LINES 23 AND 24
0097 217807    LD HL,0778     POINT TO STRING "ENTER COMMAND"
009A CDEF05    CALL 05EF      PRINT STRING "ENTER COMMAND"
009D 21E0F2    LD HL,F2E0     POINT TO LINE 24 START
00A0 363E      LD (HL),3E     PRINT THE PROMPT
00A2 23        INC HL         POINT TO NEXT POSITION
00A3 310010    LD SP,1000     (ENTRY)INITIALISE THE STACK
00A6 CBFE      SET 7,(HL)     (CONT)PRINT THE CURSOR
00A8 CDE206    CALL 06E2      (TRY AGAIN)READ THE KEYBOARD
00AB 28FB      JRZ FB         IF NO KEY JUMP TO (TRY AGAIN)
00AD FE5B      CP 5B          TEST FOR KEY [
00AF 281F      JRZ 1F         IF YES JUMP TO (CURLEFT)
00B1 FE5D      CP 5D          TEST FOR KEY ]
00B3 2811      JRZ 11         IF YES JUMP TO(CURRIGHT)
00B5 FE03      CP 03          TEST FOR KEY EXT CTRL-C
00B7 28D8      JRZ D8         IF YES JUMP TO (ENTCOM)
00B9 FE0C      CP 0C          TEST FOR KEY FF
00BB 28D1      JRZ D1         IF YES JUMP TO (CVDU)
00BD FE0D      CP 0D          TEST FOR KEY CR
00BF 2823      JRZ 23         IF YES JUMP TO (DO LINE 24)
00C1 FE08      CP 08          TEST FOR KEY BS
00C3 2815      JRZ 15         IF YES JUMP TO (BS)
00C5 77        LD (HL),A      PRINT THE KEY
00C6 7D        LD A,L         (CURIGHT)GET CURSOR POSITION
00C7 FEFF      CP FF          TEST FOR LINE END
00C9 28DB      JRZ DB         IF YES JUMP TO (CONT)
00CB CBBE      RES 7,(HL)     REMOVE THE CURSOR
00CD 23        INC HL         POINT TO NEXT POSITION
00CE 18D6      JR D6          JUMP TO (CONT)
00D0 7D        LD A,L         (CURLEFT)GET CURSOR POSITION
00D1 FEE1      CP E1          TEST FOR LINE START
00D3 28D1      JRZ D1         IF YES JUMP TO (CONT)
00D5 CBBE      RES 7,(HL)     REMOVE THE CURSOR
00D7 2B        DEC HL         POINT TO NEW POSITION
00D8 18CC      JR CC          JUMP TO (CONT)
00DA 7D        LD A,L         (BS)GET CURSOR POSITION
00DB FEE1      CP E1          TEST FOR LINE START
00DD 28C7      JRZ C7         IF YES JUMP TO (CONT)
00DF 3620      LD (HL),20     CLEAR CURRENT POSITION
00F1 2B        DEC HL         POINT TO NEXT POSITION
00E2 18C2      JR C2          JUMP TO (CONT)
00E4 CD0207    CALL 0702      (DO LINE 24)CLEAR LINE 23
00E7 23        INC HL         POINT TO COMMAND CODE
00E8 4E        LD C,(HL)      STORE IT TO C
00E9 E5        PUSH HL        STACK THIS AS LOC
```

```
00EA 21B907    LD HL,07B9     TABLE START-5
00ED 110500    LD DE,0005     ITEM LENGTH
00F0 19        ADD HL,DE      (TABLE) POINT TO FIRST/NEXT ITEM
00F1 7E        LD A,(HL)      FETCH THE ITEM
00F2 B9        CP C           DOES IT EQUATE WITH THE COMMAND?
00F3 281B      JRZ 1B         IF YES JUMP TO (FOUND)
00F5 FEFF      CP FF          TEST FOR DELIMITER?
00F7 20F7      JRNZ F7        IF NO THEN JUMP TO (TABLE)
00F9 3E58      LD A,58        LOAD ERROR CODE X
00FB 1801      JR 01          JUMP TO (ERROR 2)
00FD E5        PUSH HL        (ERROR 1) STACK THE LOCATION
00FE F5        PUSH AF        (ERROR 2) STACK THE ERROR
00FF 216307    LD HL,0768     POINT TO STRING "ERROR ("
0102 CDEF05    CALL 05EF      PRINT "ERROR ("
0105 F1        POP AF         GET THE ERROR CODE
0106 21C8F2    LD HL,F2C8     POINTER TO VDU
0109 77        LD (HL),A      PRINT THE ERROR CODE
010A 23        INC HL         POINT TO NEXT POSITION
010B 3629      LD (HL),29     PRINT ")"
010D E1        POP HL         GET THE LOCATION      00A3
010E 1893      JR 93          JUMP TO (ENTRY)
0110 0602      LD B,02        (FOUND) LOAD LOOP COUNT
0112 11DD0F    LD DE,OFDD     POINT TO SCRATCHPAD
0115 AF        XOR A          CLEAR ACC
0116 12        LD (DE),A      STORE A IN SCRATCHPAD
0117 1B        DEC DE         DECREMENT SCRATCHPAD POINTER
0118 23        INC HL         INCREMENT TABLE POINTER
0119 ED6F      RLD            LOAD EXPECTED 1ST, LENGTH 2 2ND
011B 12        LD (DE),A      STORE IN SCRATCHPAD
011C ED67      RRD            RESTORE TABLE
011E 1B        DEC DE         DECREMENT SCRATCHPAD POINTER
011F 7E        LD A,(HL)      LOAD LENGTH 1 1ST, LENGTH 3 2ND
0120 E60F      AND 0F         PASS LOWER BITS ONLY
0122 12        LD (DE),A      STORE IN SCRATCHPAD
0123 1B        DEC DE         ADJUST SCRATCHPAD POINTER
0124 10F2      DBJNZ F2       GET SECOND PART
0126 AF        XOR A          A=0
0127 12        LD (DE)A       STORE A TO SCRATCHPAD
0128 1B        DEC DE         ADJUST SCRATCHPAD POINTER
0129 23        INC HL         ADJUST TABLE POINTER
012A 23        INC HL
012B 7E        LD A,(HL)      LOAD LOW BYTE OF CMD ADDR
012C 12        LD (DE),A      SAVE IN THE SCRATCHPAD
012D 1B        DEC DE         DROP SCRATCHPAD POINTER DOWN
012E 2B        DEC HL         POINT TO HIGH BYTE OF CMD
012F 7E        LD A,(HL)      GET HIGH BYTE OF CMD ADDR
0130 12        LD (DE),A      SAVE IN THE SCRATCHPAD
0131 310010    LD SP 1000     (GET PARAMS) RESTORE SP
0134 DD21DE0F  LD IX,0FDE     POINT TO PARAM SPACE
0138 FD21DB0F  LD IY,0FDB     POINT TO DETAILS
013C 21E2F2    LD HL,F2E2     POINT TO EMBEDDED SPACE
013F E5        PUSH HL        STACK THIS AS LOC
0140 3E20      LD A,20        KEY FOR COMPARE
0142 BE        CP (HL)        TEST (LOC)=SPACE?
0143 3E42      LD A,42        ERROR CODE B
0145 20B7      JRNZ B7        IF TEST FAILED JUMP TO (ERROR 2)
0147 E1        POP HL         UNSTACK LOC
0148 23        INC HL         STEP ON
0149 E5        PUSH HL        (NEXT) RESTACK LOC
014A 3E20      LD A,20        (SCAN) KEY FOR COMPARE
014C BE        CP (HL)        DATA OR SPACE?
```

```
014D 2012      JRNZ 12          IF NOT SPACE JUMP TO (DATA)
014F 7D        LD A,L           GET LINE POSITION
0150 FEFF      CP FF            TEST FOR LINE END
0152 2803      JRZ 03           IF LINE END JUMP TO (LINE END)
0154 23        INC HL           ADVANCE THE POINTER BY ONE
0155 18F3      JR F3            JUMP BACK TO (SCAN)          (014A)
0157 3ADC0F    LD A,(0FDC)      (LINE END) GET EXPECTED
015A B7        OR A             SET THE FLAGS
015B 2849      JRZ 49           IF EXPECTED=ZERO JUMP TO (TRANSFER)
015D 3E46      LD A,46          SET UP ERROR F CODE
015F 189D      JR 9D            JUMP TO (ERROR 2)
0161 D1        POP DE           (DATA) GET LOC
0162 CDBB06    CALL 06BB        COMPARE LOC TO ACTUAL
0165 D5        PUSH DE          RESTACK LOC                  (00FE)
0166 3E50      LD A,50          SET UP FOR ERROR P
0168 20F5      JRNZ F5          IF COMPARE FAILED JUMP TO (ERROR 2)
016A CD8906    CALL 0689        GET LENGTH
016D CD4B06    CALL 064B        CHECK FOR VALID ASCII/HEX
0170 E3        EX (SP),HL       PUT LOC TO HL
0171 CD8906    CALL 0689        GET LENGTH
0174 CD9605    CALL 0596        CONVERT TO BINARY
0177 05        DEC B
0178 05        DEC B            LENGTH=LENGTH-2
0179 78        LD A,B           LENGTH TO ACC
017A B7        OR A             STATICISE THE FLAGS
017B 280B      JRZ 0B           IF LENGTH=ZERO JUMP TO (LOW)
017D DD7101    LD (IX+1),C      SAVE THE HIGH BYTE
0180 CD9605    CALL 0596        CONVERT TO BINARY
0183 DD7100    LD (IX+0),C      SAVE THE LOW BYTE
0186 1806      JR 06            JUMP TO (SET NEXT)
0188 DD7100    LD (IX+0),C      (LOW)SAVE THE LOW BYTE
018B DD7101    LD (IX+1),A      SAVE THE HIGH BYTE AS ZERO
018E DD23      INC IX           (SET NEXT)
0190 DD23      INC IX           POINT TO NEXT PARAM STORE
0192 FD2B      DEC IY           POINT TO NEXT LENGTH VALUE
0194 21DD0F    LD HL,0FDD       POINT TO FOUND
0197 34        INC (HL)         FOUND=FOUND+1
0198 3ADC0F    LD A,(0FDC)      GET EXPECTED
019B 3D        DEC A            EXPECTED=EXPECTED-1
019C F2A001    JPP 01A0         IF EXPECTED IS +VE JUMP TO (+VE)
019F AF        XOR A            EXPECTED=ZERO
01A0 32DC0F    LD (0FDC),A      (+VE)SAVE NEW EXPECTED
01A3 E1        POP HL           GET LOC
01A4 18A3      JR A3            JUMP TO (NEXT)               (0149)
01A6 2AD60F    LD HL,(0FD6)     (TRANSFER)GET COMMAND ADDRESS
01A9 D1        POP DE           RESTORE THE STACK
01AA E5        PUSH HL          STACK THE COMMAND ADDRESS
01AB C9        RET              RET EXECUTE THE COMMAND
01AC ED5BDE0F  LD DE,(0FDE)     (MODIFIY)FETCH PARAM 1
01B0 3ADD0F    LD A,(0FDD)      GET FOUND
01B3 3D        DEC A            FOUND=FOUND-1
01B4 2804      JRZ 04           IF FOUND=ZERO JUMP TO (READ)
01B6 3AE00F    LD A,(0FE0)      GET VALUE IN PARAM 2
01B9 12        LD (DE),A        STORE P2 IN P1
01BA CD7A06    CALL 06A0        (READ)DISPLAY ADDRESS AND DATA
01BD CDF006    CALL 06F0        MOVE IT TO LINE 23
01C0 CD0206    CALL 0602        SCROLL THE SCREEN
01C3 13        INC DE           POINT TO THE NEXT ADDRESS
01C4 CD7A06    CALL 067A        DISPLAY ADDRESS AND DATA
01C7 21E8F2    LD HL,F2E8       SET UP THE CURSOR POSITION
```

```
01CA C3A300     JP 00A3          BRANCH TO (ENTRY)
01CD CDE405     CALL 05E4        ZERO THE GHOST REGISTERS.
01D0 CDF006     CALL 06F0        MOVE LINE 24 TO LINE 23
01D3 CD0206     CALL 0602        SCROLL THE SCREEN
01D6 CDB205     CALL 05B2        PRINT THE GHOST REGISTERS
01D9 C39100     JP 0091          BRANCH TO (ENTCOM)
01DC CD8305     CALL 0583        TEST FOR ERROR G
01DF 2AE00F     LD HL,(0FE0)     LOAD BLOCK END
01E2 3AE20F     LD A,(0FE2)      GET VALUE TO BE WRITTEN
01E5 CDBB06     CALL 06BB        (FILL)TEST FOR HL<=>DE
01E8 12         LD (DE),A        WRITE AWAY THE VALUE
01E9 CA7405     JPZ 0574         IF DONE GO TO (TERMNONCHAINED)
01EC 13         INC DE           NOT DONE SO STEP ON
01ED 18F6       JR F6            JUMP TO (FILL)
01EF CD3506     CALL 0635        (EXECUTE) RESTORE USER OPCODE
01F2 3ADD0F     LD A,(0FDD)      GET FOUND
01F5 47         LD B,A           FOUND INTO B REGISTER
01F6 FE02       CP 02            IS FOUND EQUAL TO 2 ?
01F8 201D       JRNZ 1D          IF FOUND<>2 JUMP TO (NOT 2)
01FA 05         DEC B            FOUND=FOUND-1
01FB 2AE00F     LD HL,(0FE0)     GET THE INTENDED TRAP ADDRESS
01FE 22E40F     LD (0FE4),HL     SAVE THIS IN THE SCRATCHPAD
0201 7E         LD A,(HL)        GET THE INTENDED TRAP OPCODE
0202 32E60F     LD (0FE6),A      AND SAVE IT IN THE SCRATCHPAD
0205 AF         XOR A            SET ACC TO ZERO
0206 77         LD (HL),A        SET THE INTENDED TRAP TO ZERO
0207 BE         CP (HL)          IS THE INTENDED TRAP ZERO ?
0208 2808       JRZ 08           IF YES JUMP TO (TRY FFS)
020A 3E54       LD A,54          (NOT RAM) GET ERROR CODE T
020C 21E8F2     LD HL,F2E8       THE CURSOR POSITION FOR ERROR T
020F C3FD00     JP 00FD          BRANCH TO (ERROR 1)
0212 2F         CPL A            (TRY FFS) INVERT A TO FF
0213 77         LD (HL),A        SET THE INTENDED TRAP TO FF;RST 56
0214 BE         CP (HL)          IS THE INTENDED TRAP FF ?
0215 20F3       JRNZ F3          IF NOT JUMP TO (NOT RAM)
0217 1029       DBJNZ 29         (NOT 2) IF ITS FF JUMP TO (WAS 0)
0219 CDF006     CALL 06F0        (WAS 1) MOVE COMMAND TO LINE 23
021C CD0206     CALL 0602        SCROLL THE SCREEN
021F 21E0F2     LD HL,F2E0       POINT TO LINE 24 START
0222 CD0207     CALL 0702        CLEAR LINES 24 AND 23
0225 2ADE0F     LD HL,(0FDE)     GET EXECUTION ADDRESS
0228 22C60F     LD (0FC6),HL     SAVE IT IN THE GHOST PC
022B 3EC3       LD A,C3          GET THE UNCONDITIONAL BRANCH OPCODE.
022D 32DD0F     LD (0FDD),A      PUT IT IN FRONT OF THE EXECUTE.
0230 31CA0F     LD SP,0FCA       POINT TO THE GHOSTS
0233 DDE1       POP IX           SET USERS IX
0235 FDE1       POP IY           SET USERS IY
0237 F1         POP AF           SET USERS AF
0238 C1         POP BC           SET USERS BC
0239 D1         POP DE           SET USERS DE
023A E1         POP HL           SET USERS HL
023B ED7BC80F   LD SP,(0FC8)     SET USERS SP
023F C3DD0F     JP 0FDD          BRANCH TO THE USER'S PROGRAM
0242 CDA406     CALL 06A4        (WAS 0) SET UP COMMAND E G.PC
0245 C3A300     JP 00A3          BRANCH TO (ENTRY)
0248 CDF006     CALL 06F0        (TABULATE) MOVE LINE 24 TO 23
024B CD0206     CALL 0602        SCROLL THE SCREEN
024E 3E08       LD A,08          TAB: THE NUMBER OF LINES.
0250 ED5BDE0F   LD DE,(0FDE)     GET THE START ADDRESS
0254 F5         PUSH AF          (DO 8) SAVE THE LINE COUNT
```

```
0255 0608       LD B,08          SET THE NUMBER OF BYTES PER LINE.
0257 21C1F2     LD HL,F2C1       POINTER TO THE SCREEN
025A CD9406     CALL 0694        DISPLAY THE ADDRESS
025D 1A         LD A,(DE)        (8 BYTES). GET THE FIRST/NEXT BYTE
025E 4F         LD C,A           MOVE IT INTO THE C REGISTER
025F CD1806     CALL 0618        DISPLAY THE BYTE AS ASCIIHEX
0262 23         INC HL           STEP THE SCREEN POINTER ON
0263 13         INC DE           SET NEXT SEQUENTUAL ADDRESS
0264 10F7       DBJNZ F7         IF 8 NOT DONE JUMP TO (8 BYTES)
0266 CD0206     CALL 0602        SCROLL THE SCREEN
0296 F1         POP AF           RECOVER THE LINE COUNT
026A 3D         DEC A            COUNT = COUNT-1
026B 20E7       JRNZ E7          IF NOT DONE JUMP TO (DO 8)      -> 0254
026D 21E3F2     LD HL,F2E3       POINTER TO ON SCREEN ADDRESS
0270 CD9406     CALL 0694        DISPLAY THE NEXT SEQUENTUAL ADDRESS.
0273 C3A300     JP 00A3          AND BRANCH TO (ENTRY)           -> 00A3
0276 2AE00F     LD HL,(0FE0)     (JUMP) GET JUMP DESTINATION
0279 ED5BDE0F   LD DE,(0FDE)     GET JUMP INSTRUCTION ADDRESS
027D AF         XOR A            CLEAR THE CARRY FLAG
027E ED52       SBC HL,DE        COMPUTE THE DIFFERENCE
0280 2B         DEC HL
0281 2B         DEC HL           ADJUST FOR INSTRUCTION LENGTH
0282 4D         LD C,L           SAVE THE DISPLACEMENT FOR LATER
0283 3EFF       LD A,FF          KEY FOR COMPARE
0285 BC         CP H             TEST FOR H=FF
0286 281A       JRZ 1A           IF THE HIGH BYTE IS FF JUMP TO (H=FF)
0288 AF         XOR A            SET ACC=ZERO
0289 BC         CP H             TEST FOR H=00
028A 2808       JRZ 08           IF THE HIGH BYTE IS 00 JUMP TO (H=00)
028C 3E4A       LD A,4A          (ERR J) ERROR J CODE TO ACC
028E 21E7F2     LD HL,F2E7       ON SCREEN CURSOR FOR ERROR J
0291 C3FD00     JP 00FD          BRANCH TO (ERROR 1)             -> 00FD
0294 7D         LD A,L           (H=00) GET THE LOW BYTE OF THE DISPLACEMENT
0295 B7         OR A             SET UP THE FLAGS
0296 FA8C02     JPN 028C         IF H=00 & L IS -VE JUMP TO (ERR J)
0299 21EDF2     LD HL,F2ED       (OK) POINTER TO SCREEN FOR RESULT.
029C CD1806     CALL 0618        (PORT READ) DISPLAY THE BYTE
029F C37405     JP 0574          (EXIT) BRANCH TO (TERMNONCHAINED)   -> 0574
02A2 7D         LD A,L           (H=FF) GET THE LOW BYTE OF THE DISPLACEMENT
02A3 B7         OR A             SET UP THE FLAGS
02A4 F28C02     JPP 028C         IF H=FF & L IS +VE JUMP TO (ERR J)
02A7 18F0       JR F0            JUMP TO (OK)
02A9 3ADE0F     LD A,(0FDE)      (PORT) GET PORT ADDRESS
02AC 4F         LD C,A           SAVE IT IN C REG
02AD 3ADD0F     LD A,(0FDD)      GET FOUND
02B0 3D         DEC A            FOUND=FOUND-1
02B1 2008       JRNZ 08          IF FOUND<>0 JUMP TO (WRITE)
02B3 ED78       IN A,(C)         (READ) GET THE PORT DATA
02B5 4F         LD C,A           PUT IT INTO C
02B6 21E6F2     LD HL,F2E6       POINT TO SCREEN DESTINATION
02B9 18E1       JR E1            JUMP TO (PORT READ)
02BB 3AE00F     LD A,(0FE0)      (WRITE) GET DATA TO BE WRITTEN
02BE ED79       OUT A,(C)        WRITE THE DATA TO THE PORT
02C0 18DD       JR DD            JUMP TO (EXIT)
02C2 CD2005     CALL 0520        (VERIFY) TEST FOR ERROR G,& GET COUNT
02C5 2AE20F     LD HL,(0FE2)     GET THE START OF BLOCK 2
02C8 1A         LD A,(DE)        (NEXT) GET FIRST/NEXT BLOCK 1 DATA
02C9 1A         LD A,(DE)        GET IT AGAIN,ENSURES ADDR LATCHED OK.
02CA EDA1       CPI              COMPARE BLOCK 1/BLOCK 2 DATA
02CC 2006       JR 06            IF COMPARE FAILED JUMP TO (FAIL)  -> 02D4
```

```
02CE E24905    JPPO 0549      (DONE ?) IF ALL DONE BRANCH TO (PATCH 1)
02D1 13        INC DE         NEXT BLOCK 2 DATA
02D2 18F4      JR F4          JUMP TO (NEXT)
02D4 2B        DEC HL         (FAILS) POINT TO BLOCK 1 DATA
02D5 E5        PUSH HL        STACK IT
02D6 D5        PUSH DE        STACK BLOCK 1 DATA ALSO
02D7 23        INC HL         SET UP BLOCK 1 NEXT COMPARE
02D8 D9        EXX            TEMP SAVE HL,DE,BC
02D9 08        EX             TEMP SAVE AF
02DA 216307    LD HL,0763     POINT TO STRING "ERROR ("
02DD CDEF05    CALL 05EF      PRINT STRING "ERROR ("
02E0 1B        DEC DE
02E1 1B        DEC DE         MOVE TO OBLITERATE (
02E2 C33705    JP 0537        BRANCH TO (PATCH 4)
02E5 CD0206    CALL 0602      SCROLL THE SCREEN
02E8 08        EX             RESTORE AF
02E9 D9        EXX            RESTORE HL,DE,BC.
02EA 18E2      JR E2          JUMP TO (DONE ?)
02EC 00        NOP
02ED CD8305    CALL 0583      (COPY) TEST FOR ERROR G
02F0 EB        EX HL,DE       SOURCE START TO HL
02F1 ED5BE20F  LD DE,(0FE2)   DESTINATION START TO DE
02F5 CDBB06    CALL 06BB      TEST FOR CASE 1 COPY
02F8 301E      JRNC 1E        IF CASE 1 JUMP TO (CASE 1)
02FA EB        EX HL,DE       DESTINATION START TO HL
02FB ED5BE00F  LD DE,(0FE0)   SOURCE END TO DE
02FF CDBB06    CALL 06BB      TEST FOR CASE 2 POSSIBILITY
0302 3012      JRNC 12        IF POSSIBLE CASE 2 JUMP TO (POS CASE 2)
0304 CD2905    CALL 0529      (CASE 2) GET BYTE COUNT
0307 AF        XOR A          ZERO ACC
0308 ED5BE20F  LD DE,(0FE2)   DESTINATION START TO DE
030C ED5A      ADD HL,DE      COMPUTE DESTINATION END
030E EB        EX HL,DE       DESTINATION END TO DE
030F 2AE00F    LD HL,(0FE0)   SOURCE END TO HL
0312 EDB8      LDDR           COPY CASE 2 BLOCK 1 TO BLOCK 2.
0314 180C      JR 0C          JUMP TO (DONE)
0316 28EC      JRZ EC         (POS CASE 2) IF CASE 2 JUMP TO (CASE 2)
0318 CD2905    CALL 0529      (CASE 1) GET THE BYTE COUNT
031B EB        EX HL,DE       SOURCE START INTO HL
031C ED5BE20F  LD DE,(0FE2)   DESTINATION START TO DE
0320 EDB0      LDIR           COPY CASE 1 BLOCK 1 TO BLOCK 2.
0322 189C      JR 9C          (DONE) HOP SKIP TO (TERMNONCHAINED)
0324 CD2005    CALL 0520      (BURN) TEST FOR ERROR G, GET COUNT
0327 D5        PUSH DE        SAVE START
0328 C5        PUSH BC        SAVE COUNT
0329 3E01      LD A,01        SWITCH BYTE FOR PROGRAMMER
032B 0600      LD B,00        DELAY DURATION=256x1ms WITH 4MHZ CLK
032D D3F0      OUT A,(F0)     SWITCH ON PROGRAMMER +5 VOLTS
032F CDF204    CALL 04F2      WAIT 256x1ms=256ms;  RAMP UP POWER
0332 C1        POP BC         RECOVER COUNT
0333 D1        POP DE         RECOVER START
0334 2AE20F    LD HL,(0FE2)   GET DESTINATION START
0337 3EFF      LD A,FF        TEST BIT PATTERN ALL ONES.
0339 EDA1      CPI            (TEST) CHECK DEST BLOCK BYTE IS FF ?
033B 2005      JRNZ 05        IF NOT JUMP TO (NOT FF)
033D E26203    JPPO 0362      IF ALL TESTED OK BRANCH TO (BURN)
0340 18F7      JR F7          JUMP BACK TO (TEST)
0342 210C07    LD HL,070C     (NOT FF) POINT TO "NOT FF..." STRING
0345 11C0F2    LD DE,F2C0     POINT TO LINE 23 START
0348 CDF205    CALL 05F2      PRINT THE STRING "NOT FF....."
```

02C8 · 0549 · 0537 · 029F · 0362

```
034B CDE206    CALL 06DE      (NOK) READ THE KEY CODE.
034E FE02      CP 02          TEST FOR CTRL B
0350 2810      JRZ 10         IF CTRL B JUMP TO (BURN)
0352 FE0D      CP 0D          TEST FOR CR
0354 20F5      JRNZ F5        IF NOT CR JUMP TO (NOK)
0356 21A0F2    LD HL,F2A0     POINT TO LINE 23 START
0359 CDD706    CALL 06D7      CLEAR LINE 23
035C AF        XOR A          ZERO ACC
035D D3F0      OUT A,(F0)     SWITCH OFF THE PROGRAMMER.
035F C39100    JP 0091        BRANCH TO (ENTCOM)
0362 3E03      LD A,03        (BURN) SWITCH BYTE FOR +5 & +25 VOLTS
0364 D3F0      OUT A,(F0)     SWITCH ON +25V, MAINTAIN +5V
0366 21E0F2    LD HL,F2E0     POINT TO LINE 24 START
0369 CD0207    CALL 0702      CLEAR LINE 23 & LINE 24
036C EB        EX HL,DE       DE POINTS TO LINE 24 START
036D 214B07    LD HL,074B     HL POINTS TO STRING "BURN IN PROG"
0370 CDF205    CALL 05F2      PRINT STRING "BURN IN PROGRESS"
0373 C34F05    JP 054F        BRANCH TO (PATCH 2)
0376 00        NOP
0377 00        NOP
0378 CD2005    CALL 0520      (SAVE) TEST ERROR G; GET COUNT
037B C5        PUSH BC        STACK THE COUNT
037C 21E0F2    LD HL,F2C0     POINT TO LINE 23 START.
037F CD0207    CALL 0702      CLEAR LINE 23 & LINE 24
0382 0640      LD B,40        COUNT 64 BINARIES FOR LEADER
0384 21E1F2    LD HL,F2E1     (LEAD) POINT TO LINE 24 START+1
0387 48        LD C,B         COUNT TO THE C REGISTER FOR DISPLAY
0388 CD1806    CALL 0618      DISPLAY THE COUNT
038B AF        XOR A          DATA FOR THE LEADER
038C CDB104    CALL 04B1      TAPE SAVE ACC
038F 10F3      DBJNZ F3       IF NOT DONE JUMP TO (LEAD)
0391 3EFF      LD A,FF        DATA FOR THE TAPE MARK
0393 CDB104    CALL 04B1      TAPE SAVE MARK
0396 DDE1      POP IX         OVERALL COUNT TO IX
0398 D5        PUSH DE        STACK SOURCE START
0399 DDE5      PUSH IX        STACK OVERALL COUNT
039B 010002    LD BC,0200     B HAS 02; C HAS 00
039E AF        XOR A          (DO 2) CLEAR ACC
039F D1        POP DE         1ST TIME GET OVERALL COUNT, 2ND GET START.
03A0 82        ADD A,D        A=A+D
03A1 83        ADD A,E        A=A+E
03A2 81        ADD A,C        A=A+C
03A3 4F        LD C,A         C=D+E+C
03A4 7A        LD A,D         DATA FOR TAPE SAVE
03A5 CDB104    CALL 04B1      TAPE SAVE THE ACC
03A8 2B        DEC HL         OBLITERATE SPACE ON THE SCREEN
03A9 7B        LD A,E         DATA FOR TAPE SAVE
03AA CDB104    CALL 04B1      TAPE SAVE THE A REGISTER
03AD 10EF      DBJNZ EF       IF NOT DONE JUMP TO (DO 2)
03AF 79        LD A,C         CHECKSUM OF BB+CC+SS+AA TO ACC
03B0 CDB104    CALL 04B1      TAPE SAVE THE CHECKSUM
03B3 D5        PUSH DE        (NEXT B) STACK THE ADDRESS
03B4 0610      LD B,10        PART OF SCROLL DELAY
03B6 CDF204    CALL 04F2      DELAY 10*2MS.
03B9 D1        POP DE         RECOVER THE ADDRESS
03BA 21E1F2    LD HL,F2E1     POINT TO LINE 24 START+1
03BD CD9406    CALL 0694      DISPLAY THE ADDRESS
03C0 010008    LD BC,0800     B=BLOCK COUNT; C=CHECKSUM.
03C3 1A        LD A,(DE)      (DO 8) READ THE DATA FOR TAPE SAVE
03C4 81        ADD A,C        ADD THE CHECKSUM TO THE DATA
03C5 4F        LD C,A         SAVE THE RESULT AS THE NEW CHECKSUM
```

```
03C6 1A       LD A,(DE)     RF-READ THE DATA TO BE SAVED
03C7 CDB104    CALL 04B1     TAPE SAVE THE DATA FROM (DE)
03CA CDC904   CALL 04C9     OVERALL COUNT=OVERALL COUNT-1
03CD 2809     JRZ 09        IF ALL SAVED JUMP TO (DONE)
03CF 13       INC DE        STEP THE ADDRESS BY 1
03D0 10F1     DBJNZ F1      IF THIS BLOCK NOT DONE JUMP TO (DO 8)
03D2 79       LD A,C        BLOCK DONE SO CHECKSUM TO ACC
03D3 CDB104   CALL 04B1     TAPE SAVE THE CHECKSUM
03D6 18DB     JR DB         JUMP TO (NEXT B)
03D8 79       LD A,C        (DONE) ALL DONE, GET THE LAST CHECKSUM
03D9 CDB104   CALL 04B1     TAPE SAVE THE CHECKSUM
03DC C39100   JP 0091       BRANCH TO (ENTCOM)
03DF 218504   LD HL,0485    (LOAD) POINT TO STRING "LOAD ACTIVE"
03E2 CDEF05   CALL 05EF     PRINT STRING "LOAD ACTIVE"
03E5 0620     LD B,20       (ZERO) EXPECTED LEADER LENGTH
03E7 21E0C2   LD HL,C2E0    (TRY NEXT) POINT TO LINE 24 START
03EA CD9104   CALL 0491     READ TAPE DATA
03ED B7       OR A          SET THE FLAGS
03EE 20F5     JRNZ F5       IF NOT A ZERO BYTE JUMP TO (ZERO)
03F0 48       LD C,B        LEADER COUNT REMAINING TO C REG
03F1 CD1806   CALL 0618     DISPLAY THE REMAINDER
03F4 10F1     DBJNZ F1      IF COUNT<>ZERO JUMP TO (TRY NEXT)
03F6 21E1F2   LD HL,F2E1    (MARK) POINT TO LINE 24 START
03F9 CD9104   CALL 0491     READ TAPE DATA
03FC FEFF     CP FF         TEST FOR MARK
03FE 20F6     JRNZ F6       IF NOT FF JUMP TO (MARK)
0400 21E0F2   LD HL,F2E0    POINT TO LINE 24 START
0403 CD0207   CALL 0702     CLEAR LINE 23 & LINE 24
0406 23       INC HL        ADVANCE THE SCREEN POINTER
0407 010002   LD BC,0200    B HAS 02, C HAS 00
040A 1801     JR 01         JUMP TO (HOP)
040C D5       PUSH DE       (DO 2) STACK THE RESULT
040D CD9104   CALL 0491     (HOP) READ TAPE DATA
0410 57       LD D,A        SAVE IT IN D REG
0411 81       ADD A,C       ADD THE CHECKSUM TO THE DATA
0412 4F       LD C,A        SAVE THE RESULT AS THE NEW CHECKSUM
0413 2B       DEC HL        MOVE THE SCREEN POINTER BACK ONCE
0414 CD9104   CALL 0491     READ TAPE DATA
0417 5F       LD E,A        SAVE IT IN THE E REG
0418 81       ADD A,C       ADD THE CHECKSUM TO THE DATA
0419 4F       LD C,A        SAVE THE RESULT AS THE NEW CHECKSUM
041A 10F0     DBJNZ F0      IF NOT DONE JUMP TO (DO 2)
041C DDE1     POP IX        OVERALL COUNT TO IX
041E CD9104   CALL 0491     READ TAPE DATA
0421 B9       CP C          DO A CHECKSUM COMPARE
0422 CA6405   JPZ 0564      ALL OK SO BRANCH TO (PATCH 3)
0425 3E43     LD A,43       ERROR C CODE
0427 C3FD00   JP 00FD       BRANCH TO (ERROR 1)
042A 21E1F2   LD HL,F2E1    (BLOCK) POINT TO LINE 24 START
042D CD9406   CALL 0694     DISPLAY THE ADDRESS
0430 010008   LD BC,0800    B HAS BLOCK ;C HAS CHECKSUM INITIAL
0433 CD9104   CALL 0491     (DO 8) READ TAPE DATA
0436 12       LD (DE),A     STORE THE DATA DESTINATION ADDRESS
0437 81       ADD A,C       ADD THE CHECKSUM TO THE DATA
0438 4F       LD C,A        SAVE THIS AS THE NEW CHECKSUM
0439 CDC904   CALL 04C9     OVERALL COUNT=OVERALL COUNT-1
043C 2811     JRZ 11        IF ALL IN JUMP TO (ALL DONE)
043E 13       INC DE        ADVANCE THE ADDRESS POINTER
043F 10F2     DBJNZ F2      IF THIS BLOCK NOT DONE JUMP TO (DO 8)
0441 CD9104   CALL 0491     READ TAPE DATA.
```

03C3 → (DO 8)
0383 → (NEXT B)
0091 → (ENTCOM)
0564 → (PATCH 3)
00FD → (ERROR 1)
044F → (ALL DONE)

```
0444 B9        CP C          DO A CHECKSUM COMPARE
0445 28E3      JRZ E3        IF OK JUMP TO (BLOCK)
0447 CDF006    CALL 06F0     MOVE LINE 24 TO LINE 23
044A CD0206    CALL 0602     SCROLL THE SCREEN
044D 18DB      JR DB         JUMP TO (BLOCK)
044F CD9104    CALL 0491     (ALL DONE) READ TAPE DATA
0452 B9        CP C          DO A CHECKSUM COMPARE
0453 2809      JRZ 09        IF OK JUMP TO (TIDY UP)
0455 CDD706    CALL 06D7     CLEAN UP THE SHORT BLOCK LINE
0458 CDF006    CALL 06F0     MOVE LINE 24 TO LINE 23
045B CD0206    CALL 0602     SCROLL THE SCREEN
045E 21E0F2    LD HL,F2E0    (TIDY UP) POINT TO LINE 24 START
0461 CD0207    CALL 0702     CLEAR LINE 24 & LINE 23
0464 23        INC HL        ADVANCE THE SCREEN POINTER
0465 364C      LD (HL),4C    PRINT "L"
0467 23        INC HL
0468 23        INC HL        ADVANCE THE SCREEN POINTER BY 2
0469 D5        PUSH DE       STACK THE END ADDRESS
046A C1        POP BC        GET THE END ADDRESS TO BC
046B D1        POP DE        GET THE START ADDRESS TO DE
046C CD9406    CALL 0694     DISPLAY THE START ADDRESS
046F C5        PUSH BC
0470 D1        POP DE        END ADDRESS TO DE
0471 CD9406    CALL 0694     DISPLAY THE END ADDRESS
0474 C37405    JP 0574       BRANCH TO (NONCHAINEDEXIT)
0477 D5        PUSH DE       (PAUSE N PRINT *SUB*) SAVE DE
0478 E5        PUSH HL       SAVE HL
0479 C5        PUSH BC       SAVE BC
047A 0613      LD B,13       THE REST OF THE SCROLL TIME DELAY
047C CDF204    CALL 04F2     DELAY B*2MS AT 2MHZ
047F C1        POP BC        RESTORE BC
0480 E1        POP HL        RESTORE HL
0481 D1        POP DE        RESTORE DE
0482 C31806    JP 0618       BRANCH TO (BIN-ASCHEX *SUB*)
0485 4C4F4144  LOAD          STRING "LOAD ACTIVE"
0489 20414354   ACT
048D 495645FF  IVE
0491 C5        PUSH BC       (FETCH A *SUB*) SAVE BC
0492 DB04      IN A,(04)     (DAV) READ TAPE STATUS
0494 0000      NOP,NOP       NOP, NOP
0496 0000      NOP,NOP       NOP, NOP
0498 CB77      BIT 6,A       IS DAV SET ?
049A 28F6      JRZ F6        IF NOT JUMP TO (DAV)
049C DB05      IN A,(05)     READ TAPE DATA
049E F5        PUSH AF       STACK TAPE DATA
049F 4F        LD C,A        DATA TO C REG FOR DISPLAY
04A0 CD1806    CALL 0618     DISPLAY THE DATA TO THE SCREEN
04A3 3620      LD (HL),20    THEN DISPLAY A SPACE
04A5 23        INC HL        AND STEP ON ONE
04A6 F1        POP AF        RESTORE THE DATA
04A7 C1        POP BC        RESTORE BC
04A8 C9        RET           RETURN ACC HAS TAPE DATA
04A9 FFFF      - -           SPARE
04AB FFFFFF    - - -         SPARE
04AE FFFFFF    - - -         SPARE
```

042A

0574

0618

```
04B1 C5          PUSH BC          (SAVE A *SUB*) SAVE BC
04B2 F5          PUSH AF          STACK THE DATA TO BE SAVED
04B3 DB04        IN A,(04)        (STATUS) GET TAPE STATUS
04B5 0000        NOP,NOP          NOP, NOP
04B7 0000        NOP,NOP          NOP, NOP
04B9 CB7F        BIT 7,A          TEST FOR TBMT
04BB 28F6        JRZ F6           IF NOT EMPTY JUMP TO (STATUS)
04BD F1          POP AF           RECOVER DATA TO BE SAVED
04BE D305        OUT A,(05)       SEND THE DATA TO TAPE
04C0 4F          LD C,A           DATA TO THE PRINT REGISTER
04C1 CD7704      CALL 0477        PAUSE AND THEN PRINT THE DATA
04C4 3620        LD (HL),20       PRINT A SPACE
04C6 23          INC HL           STEP THE SCREEN POINTER BY ONE
04C7 C1          POP BC           RESTORE BC
04C8 C9          RET              RETURN DATA TAPE SAVED.
04C9 E5          PUSH HL          (DEC IX *SUB*) SAVE HL
04CA D5          PUSH DE          SAVE DE
04CB DDE5        PUSH IX
04CD E1          POP HL           MOVE IX TO HL FOR DECREMENTING
04CE 110100      LD DE,0001       THE DECREMENT AMOUNT
04D1 AF          XOR A            CLEAR THE FLAGS
04D2 ED52        SBC HL,DE        DECREMENT THE "IX" VALUE
04D4 E5          PUSH HL
04D5 DDE1        POP IX           PUT THE RESULT BACK IN IX
04D7 D1          POP DE           RESTORE DE
04D8 E1          POP HL           RESTORE HL
04D9 C9          RET              EXIT; IX=IX-1 & THE FLAGS ARE SET UP
04DA F5          PUSH AF          (BURN IT *SUB*) SAVE A
04DB C5          PUSH BC          SAVE BC
04DC D5          PUSH DE          SAVE DE
04DD E5          PUSH HL          SAVE HL
04DE 00          NOP
04DF 00          NOP
04E0 0632        LD B,32          LOAD B WITH PULSE DURATION.
04E2 3E07        LD A,07          SWITCH BYTE FOR PROGRAMMING
04E4 D3F0        OUT A,(F0)       SWITCH ON THE PULSE BIT 2,PORT F0
04E6 CDF204      CALL 04F2        DELAY B x 1ms AT 4MHZ = 50ms
04E9 3E03        LD A 03          SWITCH BYTE TO STOP BURNING
04EB D3F0        OUT A,(F0)       SWITCH OFF THE PULSE
04ED E1          POP HL           RESTORE HL
04EE D1          POP DE           RESTORE DE
04EF C1          POP BC           RESTORE BC
04F0 F1          POP AF           RESTORE AF
04F1 C9          RET              EXIT DATA BURNT
04F2 110100      LD DE,0001       (DELAY B *SUB*) VAL FOR DECREMENT
04F5 219000      LD HL 0090       (NEXT B) HL=COUNT FOR 1ms AT 4MHZ
04F8 ED52        SBC HL,DE        (DEC) HL=HL-1 & SET FLAGS
04FA 20FC        JRNZ FC          IF HL<>0 JUMP TO (DEC)
04FC 10F7        DBJNZ,F7         B=B-1;IF B<>0 JUMP TO (NEXT B)
04FE C9          RET              EXIT DELAY OVER
04FF 21E1F2      LD HL,F2E1       (AUTO-VERIFY) LINE 24 START+1
0502 3656        LD (HL),56       PRINT V TO LINE 24 POS 2
0504 23          INC HL           POINT TO LINE 24 POS 3
0505 3620        LD (HL),20       PRINT A SPACE TO POS 3
0507 23          INC HL           POINT TO POS 4
0508 ED5BDE0F    LD DE,(0FDE)     GET THE BURN SOURCE START PARAM
050C CD9406      CALL 0694        DISPLAY THIS STARTING AT POS 4
050F ED5BE00F    LD DE,(0FE0)     GET THE BURN SOURCE END PARAM
0513 CD9406      CALL 0694        DISPLAY IT TO LINE 24
0516 ED5BE20F    LD DE,(0FE2)     GET THE BURN DESTINATION START
051A CD9406      CALL 0694        DISPLAY IT TO LINE 24
051D C3E400      JP 00E4          VERIFY UP. BRANCH TO (DO LINE 24) → ØØE4
```

```
0520 CD8305    CALL 0583       (TEST ERR G GET COUNT *SUB*)
0523 CDF006    CALL 06F0       MOVE LINE 24 TO LINE 23
0526 CD0206    CALL 0602       SCROLL THE SCREEN
0529 2AE00F    LD HL,(0FE0)    (GET COUNT *SUB*) GET SOURCE END
052C ED5BDE0F  LD DE,(0FDE)    GET SOURCE START
0530 AF        XOR A           CLEAR THE FLAGS
0531 ED52      SBC HL,DE       COMPUTE SE-SS=COUNT-1
0533 E5        PUSH HL
0534 C1        POP BC          MOVE THIS TO BC
0535 03        INC BC          BC HAS BYTE COUNT
0536 C9        RET             EXIT BYTE COUNT IN BC
0537 EB        EX HL,DE        (PATCH 4) HL HAS SCREEN POINTER
0538 D1        POP DE          ADDRESS TO DE
0539 CD7D06    CALL 067D       DISPLAY THIS ADDRESS+DATA
053C 21CEF2    LD HL,F2CE      INTER ADDRESS GAP
053F 363A      LD (HL),3A      PRINT : AS A MARKER
0541 23        INC HL          POINT TO NEXT POSITION
0542 D1        POP DE          GET THE NEXT ADDRESS
0543 CD7D06    CALL 067D       DISPLAY THE ADDRESS
0546 C3E502    JP 02E5         RETURN TO VERIFY DRIVER.
0549 AF        XOR A           (PATCH 1) SET A TO ZERO
054A D3F0      OUT A,(F0)      SWITCH OFF THE PROGRAMMER
054C C39100    JP 0091         BRANCH TO (ENTCOM)
054F CD2905    CALL 0529       (PATCH 2) GET COUNT
0552 2AE20F    LD HL,(0FE2)    GET THE DESTINATION ADDRESS
0555 EB        EX HL,DE        DE=DEST START,HL=SOURCE START
0556 EDA0      LDI             (NEXT) DATA FROM (HL) TO EPROM
0558 CDDA04    CALL 04DA       BURN THE DATA INTO THE EPROM
055B EA5605    JPPE 0556       IF NOT DONE JUMP TO (NEXT)
055E 3E01      LD A,01         SWITCH BYTE FOR PROGRAMMER
0560 D3F0      OUT A,(F0)      SWITCH OFF 25V LEAVE ON 5V
0562 189B      JR 9B           JUMP TO (AUTO-VERIFY)
0564 3ADD0F    LD A,(0FDD)     (PATCH 3) GET FOUND
0567 3D        DEC A           FOUND=FOUND-1
0568 2004      JRNZ 04         IF FOUND<>0 JUMP TO (DEFAULT)
056A ED5BDE0F  LD DE,(0FDE)    GET PARAMETER 1 TO DE
056E D5        PUSH DE         (DEFAULT) STACK THE LOAD POINT ADDR
056F C32A04    JP 042A         BRANCH TO (BLOCK)
0572 00        NOP
0573 00        NOP
0574 CDF006    CALL 06F0       (TERMNONCHAINED) MOVE 24 TO 23
0577 CD0206    CALL 0602       SCROLL THE SCREEN
057A 21E0F2    LD HL,F2E0      POINT TO LINE 24
057D CD0207    CALL 0702       CLEAR LINES 23 & 24
0580 C39100    JP 0091         BRANCH TO (ENTCOM)
0583 2AE00F    LD HL,(0FE0)    (TEST ERROR G) GET SOURCE END
0586 ED5BDE0F  LD DE,(0FDE)    GET SOURCE START
058A CDBB06    CALL 06BB       TEST FOR SS<=>SE
058D D0        RET NC          IF SS<=SE OK SO RETURN NOT ERROR G
058E 3E47      LD A,47         (ERROR G) GET CODE G TO ACC
0590 21E7F2    LD HL,F2E7      SET THE CURSOR POS FOR ERR G
0593 C3FD00    JP 00FD         BRANCH TO (ERROR 1)
```

0091
4FF
42A
FD

```
0596 0E00       LD C,00          (ASC/HEX TO BIN *SUB*) CLEAR C REG
0598 7E         LD A,(HL)        GET THE FIRST ASC/HEX DIGIT
0599 CD9D05     CALL 059D        CONVERT THIS 4 BITS IN HIGH C
059C 7E         LD A,(HL)        GET THE SECOND ASC/HEX DIGIT
059D CB21       SLA C
059F CB21       SLA C
05A1 CB21       SLA C
05A3 CB21       SLA C            SHIFT RESULT LEFT FOUR BITS=RRRR0000
05A5 D630       SUB A,30         CONVERT TO 0-15
05A7 FE10       CP 10            TEST FOR A-F
05A9 FAAE05     JPN 05AE         IF NOT A-F JUMP TO (FORM)
05AC D607       SUB A,07         CHANGE A-F TO DEC 0-15
05AE 81         ADD A,C          (FORM) ASSEMBLE THE RESULT
05AF 4F         LD C,A           SAVE THE RESULT IN C
05B0 23         INC HL           ADVANCE POINTER
05B1 C9         RET              FIRST TO 059C,SECOND EXIT DONE
05B2 218607     LD HL,0786       (PRINT REGS *SUB*) STRING PCSPIXIY
05B5 CDEF05     CALL 05EF        PRINT STRING PC SP IX IY TO LINE 23
05B8 DD21C60F   LD IX,0FC6       POINT TO GHOST REGISTERS
05BC CDC905     CALL 05C9        PRINT THE FIRST 4 AND SCROLL
05BF 21A207     LD HL,07A2       POINT TO STRING AF BC DE HL
05C2 CDEF05     CALL 05EF        PRINT STRING AF BC DE HL
05C5 CDC905     CALL 05C9        PRINT THE NEXT 4 AND SCROLL
05C8 C9         RET              EXIT DONE
05C9 0604       LD B,04          (PRINT 4 & SCROLL *SUB*) GET COUNT
05CB 21C4F2     LD HL,F2C4       DESTINATION POINTER
05CE DD5E00     LD E,(IX+0)      (DO 4) GET FIRST/NEXT LOW BYTE
05D1 DD5601     LD D,(IX+1)      GET FIRST/NEXT HIGH BYTE
05D4 CD9406     CALL 0694        DISPLAY THE 16 BITS AS ASC/HEX
05D7 DD23       INC IX
05D9 DD23       INC IX           POINT TO NEXT
05DB 23         INC HL
05DC 23         INC HL
05DD 23         INC HL           POINT TO NEXT DESTINATION
05DE 10EE       DBJNZ EE         IF 4 NOT DONE JUMP TO (DO 4)
05E0 CD0206     CALL 0602        SCROLL THE SCREEN
05E3 C9         RET              EXIT DONE
05E4 21C60F     LD HL,0FC6       (ZERO GHOSTS) POINT TO GHOST STORE
05E7 0610       LD B,10          SET THE GHOST STORE SIZE
05E9 3600       LD (HL),00       (DO 16) CLEAR THE FIRST/NEXT GHOST
05EB 23         INC HL           ADVANCE THE GHOST POINTER
05EC 10FB       DBJNZ FB         IF NOT DONE JUMP TO (DO 16)
05EE C9         RET              EXIT DONE
05EF 11C1F2     LD DE,F2C1       (PRINTSTRING 1 *SUB*) DESTINATION POINTER
05F2 3EFF       LD A,FF          (PRINTSTRING 2 *SUB*) STRING DELIMIT
05F4 BE         CP (HL)          (NEXT (HL)) TEST FOR STRING DELIMITER
05F5 2804       JRZ 04           IF DELIMIT SEEN JUMP TO (END)
05F7 EDA0       LDI              PRINT (HL) TO (DE),INC HL & DE
05F9 18F9       JR F9            JUMP TO (NEXT (HL))
05FB D5         PUSH DE          (END) STACK THE NEXT ON SCREEN POS
05FC E1         POP HL           UNSTACK TO HL THE ON SCREEN POS
05FD CDD706     CALL 06D7        CLEAR FROM LAST POS TO LINE END
0600 13         INC DE           DE HAS ON VDU STRING END+1
0601 C9         RET              EXIT,DE SET UP FOR ANOTHER STRING
```

```
0602 C5        PUSH BC        (SCROLL 1-23 & CLEAR 23 *SUB*) SAVE BC
0603 D5        PUSH DE        SAVE DE
0604 E5        PUSH HL        SAVE HL
0605 2120F0    LD HL,F020     POINT TO LINE 2
0608 1100F0    LD DE,F000     POINT TO LINE 1
060B 01C002    LD BC,02C0     THE NUMBER OF BYTES TO MOVE
060E EDB0      LDIR           SCROLL SCREEN 1-23
0610 EB        EX HL,DE       POINT TO LINE 23 START=F2C0
0611 CDD706    CALL 06D7      CLEAR LINE 23
0614 E1        POP HL         RESTORE HL
0615 D1        POP DE         RESTORE DE
0616 C1        POP BC         RESTORE BC
0617 C9        RET            EXIT DONE
0618 3EF0      LD A,F0        (BIN TO ASC/HEX *SUB*) GET MASK
061A A1        AND A,C        MASK FOR UPPER BITS OF DATA IN C
061B CB07      RRC A
061D CB07      RRC A
061F CB07      RRC A
0621 CB07      RRC A          SHIFT UPPER BITS TO LOW NIBBLE
0623 CD2906    CALL 0629      CONVERT BIN IN A TO ASC/HEX AT (HL)
0626 3E0F      LD A,0F        GET MASK
0628 A1        AND A,C        MASK FOR LOW BITS OF DATA IN C
0629 C630      ADD A,30       CONVERT TO 0-15
062B FE3A      CP 3A          TEST FOR 0-9 OR 10-15
062D FA3206    JPN 0632       IF NOT 10-15 JUMP TO (SKIP)
0630 C607      ADD A,07       MAKE 10-15 INTO A-F
0632 77        LD (HL),A      (SKIP) ,PRINT THE ASC/HEX DIGIT TO HL
0633 23        INC HL         ADVANCE THE SCREEN POINTER
0634 C9        RET            FIRST TO 0626,SECOND EXIT DONE
0635 2AE40F    LD HL,(0FE4)   (RESTORE TRAP *SUB*) GET TRAP ADDR
0638 3AE60F    LD A,(0FE6)    GET THE USER OP-CODE
063B 77        LD (HL),A      RESTORE THE USER'S OP-CODE
063C 210000    LD HL,0000     NO TRAP SET ADDRESS VALUE
063F 22E40F    LD (0FE4),HL   TRAP ADDRESS STORE TO ZERO
0642 3E21      LD A,21        NO TRAP OP-CODE VALUE
0644 32E60F    LD (0FE6),A    SAVE THE NO TRAP OP-CODE
0647 77        LD (HL),A      RESTORE ZYMON ADDRESS 0000
0648 C9        RET            EXIT,RESTORED
0649 00        NOP
064A 00        NOP
```

THIS PAGE HAS BEEN SHORT PRINTED TO AVOID SPLITTING A SUBROUTINE ACROSS TWO PAGES.

```
064B C5        PUSH BC        (CHEX *SUB*) SAVE BC
064C 012F0A    LD BC,0A2F     B HAS COUNT; C HAS FIRST-1
064F 7E        LD A,(HL)      GET THE CHARACTER
0650 0C        INC C          (0-9) SET FIRST/NEXT COMPARE CODE
0651 B9        CP C           IS THE CHARACTER AS THE C REG?
0652 2819      JRZ 19         IF YES JUMP TO (OK)
0654 10FA      DBJNZ FA       TEST ALL TRIED;IF NO JUMP TO (0-9)
0656 014006    LD BC,0640     B HAS COUNT;C HAS FIRST-1
0659 0C        INC C          (A-F) SET FIRST/NEXT COMPARE CODE
065A B9        CP C           IS THE CHARACTER AS THE C REG?
065B 2810      JRZ 10         IF YES JUMP TO (OK)
065D 10FA      DBJNZ FA       TEST ALL TRIED;IF NO JUMP TO (A-F)
065F 3E20      LD A,20        SPACE CODE FOR COMPARE
0661 BE        CP (HL)        TEST IF CHARACTER IS SPACE
0662 2805      JRZ 05         IF YES JUMP TO (SHORT)
0664 3E48      LD A,48        ERROR H CODE;NOT HEX
0666 C3FD00    JP 00FD        (ERR) BRANCH TO (ERROR 1)
0669 3E53      LD A,53        (SHORT) ERROR S;PARAM IS SHORT
066B 18F9      JR F9          JUMP TO (ERR)
066D C1        POP BC         (OK) RESTORE BC
066E 23        INC HL         ADVANCE THE SCREEN POINTER
066F 10DA      DBJNZ DA       IF ALL NOT DONE JUMP TO (CHEX)
0671 3E20      LD A,20        SPACE CODE FOR COMPARE
0673 BE        CP C           TEST FOR POST PARAM SPACE CODE?
0674 3E4C      LD A,4C        ERROR L CODE,JUST IN CASE
0676 20EE      JRNZ EE        JUMP NOT SPACE TO (ERR)
0678 23        INC HL         ADVANCE THE SCREEN POINTER
0679 C9        RET            EXIT DONE

067A 21E3F2    LD HL,F2E3     (DISP DE (DE) 1 *SUB*) POINT POS 4
067D CD9406    CALL 0694      (DISP DE (DE) 2 *SUB*) DISP DE HEX
0680 1A        LD A,(DE)      GET THE ADDRESS CONTENTS
0681 4F        LD C,A         PUT IT INTO C REG
0682 CD1806    CALL 0618      PRINT THE C REG CONTENTS AS HEX
0685 CDD706    CALL 06D7      CLEAR TO THE LINE END
0688 C9        RET            EXIT;DONE.
0689 FD4600    LD B,(IY+0)    (LD B LENGTH *SUB*) LD B FROM S/P
068C AF        XOR A          ZERO ACC
068D B8        CP B           TEST FOR B=0?
068E C0        RET NZ         RETURN IF B<>0
068F 3E4D      LD A,4D        ERROR M CODE TO ACC
0691 C3FD00    JP 00FD        BRANCH TO (ERROR 1)

0694 D5        PUSH DE        (DISP DE HEX *SUB*) SAVE DE
0695 C5        PUSH BC        SAVE BC
0696 4A        LD C,D         GET UPPER BYTE TO BE PRINTED
0697 CD1806    CALL 0618      PRINT THE BYTE IN C
069A 4B        LD C,E         GET LOWER BYTE TO BE PRINTED
069B CD1806    CALL 0618      PRINT THE BYTE IN C
069E 3620      LD (HL),20     PRINT A FOLLOWING SPACE
06A0 23        INC HL         ADVANCE THE SCREEN POINTER
06A1 C1        POP BC         RESTORE BC
06A2 D1        POP DE         RESTORE DE
06A3 C9        RET            EXIT DONE
```

ØØFD

ØØFD

```
06A4 21E0F2    LD HL,F2E0     (E PC *SUB*) POINT TO LINE 24
06A7 E5        PUSH HL        STACK THIS POINTER
06A8 CDD706    CALL 06D7      CLEAR TO LINE END
06AB E1        POP HL         RESTORE POINTER
06AC 363E      LD (HL),3E     PRINT THE PROMPT
06AE 23        INC HL         ADVANCE THE POINTER
06AF 3645      LD (HL),45     PRINT "E"
06B1 23        INC HL
06B2 23        INC HL         ADVANCE THE POINTER
06B3 ED5BC60F  LD DE,(0FC6)   FETCH THE GHOST PC
06B7 CD9406    CALL 0694      PRINT IT TO LINE 24
06BA C9        RET            EXIT DONE.
06BB E5        PUSH HL        (HL-DE *SUB*) SAVE HL
06BC 37        SCF            SET THE CARRY FLAG
06BD 3F        CCF            COMPLEMENT THE CARRY FLAG = 0
06BE ED52      SBC HL,DE      SUBTRACT DE FROM HL
06C0 00        NOP
06C1 E1        POP HL         RESTORE HL
06C2 C9        RET            EXIT;HL=DE Z=1;HL>DE ALL=0;HL<DE S=1,C=1
06C3 C5        PUSH BC        (CVDU *SUB*) SAVE BC
06C4 D5        PUSH DE        SAVE DE
06C5 E5        PUSH HL        SAVE HL
06C6 2100F0    LD HL,F000     POINT TO SCREEN START
06C9 1101F0    LD DE,F001     POINTER TO START+1
06CC 010003    LD BC,0300     SCREEN SIZE
06CF 3620      LD (HL),20     CLEAR THE FIRST LOCATION
06D1 EDB0      LDIR           CLEAR THE REST
06D3 E1        POP HL         RESTORE HL
06D4 D1        POP DE         RESTORE DE
06D5 C1        POP BC         RESTORE BC
06D6 C9        RET            EXIT DONE
06D7 F5        PUSH AF        (CLINEND *SUB*) SAVE AF
06D8 3620      LD (HL),20     (CONT) PRINT A SPACE TO (HL)
06DA 23        INC HL         ADVANCE THE POINTER
06DB 7D        LD A,L         GET LOW BYTE OF POINTER
06DC E61F      AND 1F         MASK IT FOR LOWER 5 BITS(LINE)
06DE 20F8      JRNZ F8        IF NOT DONE JUMP TO (CONT)
06E0 F1        POP AF         RESTORE ACC
06E1 C9        RET            EXIT DONE
06E2 1600      LD D,00        (KBR *SUB*) ZERO TEMP STORE
06E4 DB40      IN A,(40)      (GETKEY) READ KEYCODE
06E6 C680      ADD A,80       STROBE TO CARRY FLAG
06E8 3003      JRNC 03        IF NO STROBE JUMP TO (NOS)
06EA 57        LD D,A         KEYCODE TO TEMP STORE
06EB 18F7      JR F7          JUMP TO (GETKEY)
06ED 7A        LD A,D         (NOS) TEMPSTORE TO ACC
06EE B7        OR A           STATICISE THE FLAGS
06EF C9        RET            RETURN KEY IN A; Z FLAG IF VALID KEY
06F0 E5        PUSH HL        (MV24-23 *SUB*) SAVE HL
06F1 D5        PUSH DE        SAVE DE
06F2 C5        PUSH BC        SAVE BC
06F3 21E1F2    LD HL,F2E1     POINT TO LINE 24
06F6 11C1F2    LD DE,F2C1     POINT TO LINE 23
06F9 011F00    LD BC,001F     LINE LENGTH
06FC EDB0      LDIR           MOVE 24-23
06FE C1        POP BC         RESTORE BC
06FF D1        POP DE         RESTORE DE
0700 E1        POP HL         RESTORE HL
0701 C9        RET            EXIT DONE
```

```
0702 CDD706    CALL 06D7       (CLHL+23 *SUB*) CLEAR HL TO LINEND
0705 21C0F2    LD HL,F2C0      POINT TO LINE 23 START
0708 CDD706    CALL 06D7       CLEAR HL TO LINEND
070B C9        RET             EXIT DONE
-------------------------------------------------------------------------
070C 54484520  THE             STRING "THE DESTINATION IS NOT ALL FFS!
0710 44455354  DEST                    TO CONTINUE,CTRL+B.TO QUIT,CR."
0714 494E4154  INAT
0718 494F4E20  ION
071C 4953204E  IS N
0720 4F542041  OT A
0724 4C4C2046  LL F
0728 46275321  F"S!
072C 544F2043  TO C
0730 4F4E5449  ONTI
0734 4E55452C  NUE,
0738 4354524C  CTRL
073C 2B422E54  +B.T
0740 4F205155  O QU
0744 49542C43  IT,C
0748 522EFF    R.
-------------------------------------------------------------------------
074B 4550524F  EPRO            STRING "EPROM BURN IN PROGRESS"
074F 4D204255  M BU
0753 524E2049  RN I
0757 4E205052  N PR
075B 4F475245  OGRE
075F 53532EFF  SS.
-------------------------------------------------------------------------
0763 4552524F  ERRO            STRING "ERROR ("
0767 522028FF  R (
-------------------------------------------------------------------------
076B 5A594D4F  ZYMO            STRING "ZYMON 2.VXXX"
076F 4E20322E  N X.
0773 56XXXXXX  VXXX
0777 FF
-------------------------------------------------------------------------
0778 454E5445  ENTE            STRING "ENTER COMMAND"
077C 5220434F  R CO
0780 4D4D414E  MMAN
0784 44FF      D
-------------------------------------------------------------------------
0786 50433D20  PC=             STRING "PC=    SP=    IX=    IY="
078A 20202020
078E 53503D20  SP=
0792 20202020
0796 49583D20  IX=
079A 20202020
079E 49593DFF  IY=
-------------------------------------------------------------------------
07A2 41463D20  AF=             STRING "AF=    BC=    DE=    HL="
07A6 20202020
07AA 42433D20  BC=
07AE 20202020
07B2 44453D20  DE=
07B6 20202020
07BA 484C3DFF  HL=
```

COMMAND TABLE

```
07BE 4D1420AC01 MODIFY    M:1:4:2:0:01AC
07C3 5414004802 TABULATE  T:1:4:0:0:024B
07C8 433444ED02 COPY      C:3:4:4:4:02FD
07CD 463442DC01 FILL      F:3:4:4:2:01DC
07D2 5324407803 SAVE      S:2:4:4:0:0378
07D7 4C0400DF03 LOAD      L:0:4:0:0:03DF
07DC 450440EF01 EXECUTE   E:0:4:4:0:01EF
07E1 4A24407602 JUMP      J:2:4:4:0:0276
07E6 501220A902 PORT      P:1:2:2:0:02A9
07EB 4234442403 BURN      B:3:4:4:4:0324
07F0 563444C202 VERIFY    V:3:4:4:4:02C2
07F5 5A0000CD01 ZERO      Z:0:0:0:0:01CD
07FA 520000D001 REGISTERS R:0:0:0:0:01D0
07FF FF         TABLE DELIMIT.
```

TABLE ENTRIES ARE HELD AS:-

ASCII COMMAND LETTER
MANDATORY NUMBER OF PARAMETERS
THE NUMBER OF CHARACTERS IN THE FIRST
THE NUMBER OF CHARACTERS IN THE SECOND
THE NUMBER OF CHARACTERS IN THE THIRD
THE ADDRESS OF THE COMMAND DRIVER ROUTINE

E.G.:-

M:1:4:2:0:01AC

M=COMMAND LETTER
1=MINIMUM NUMBER OF MANDATORY PARAMETERS
4=THE LENGTH OF THE FIRST PARAMETER
2=THE LENGTH OF THE SECOND PARAMETER
0=THE LENGTH OF THE THIRD PARAMETER (THERE ISN'T ONE IN THIS CASE)
01AC=THE ADDRESS OF THE COMMAND DRIVER ROUTINE

END OF LISTING FOR ZYMON 2

USER OPTION

THE KEY TO ACTION MAY BE ALTERED AT WILL (SEE PAGE 16). CURRENTLY THE KEYS ARE:

ADDR	KEYCODE		FUNCTION
00AE	5B	[	CURSOR LEFT (NON-DESTRUCTIVE)
00B2	5D	]	CURSOR RIGHT (NON-DESTRUCTIVE)
00B6	03	EXT	(CTRL-C) INPUT COMMAND
00BE	0D	CR	OBEY THE LINE 24 COMMAND
00C2	08	BS	BACKSPACE (DESTRUCTIVE CURSOR LEFT)

IF YOU DO NOT HAVE THE REQUIRED KEY ON YOUR KEYBOARD YOU CAN SUBSTITUTE ANY MORE CONVENIENT KEY YOURSELF. SOME PEOPLE PREFER "ESCAPE" (1B) AS THE CONTENTS OF 00B6 IN PLACE OF "EXT" (03); ANOTHER EXAMPLE WOULD BE THE USE OF "DELETE" (7F) AT ADDRESS 00C2, IN PLACE OF "BACKSPACE" (08), IF THE KEYBOARD USED HAS A "DELETE" KEY BUT NO "BACKSPACE".

- SEE PAGE 32b (NEXT) FOR USER CUSTOMISATION OF SAVE COMMAND -

USER OPTION FOR SAVE COMMAND

THE SAVE COMMAND IS DESCRIBED ON PAGE 11 OF THIS MANUAL. THERE IS SOME SCOPE FOR YOU TO ALTER ZYMON TO SUIT YOUR OWN PERSONAL THOUGHTS ON ERROR DETECTION. FIRST READ THE BRIEF HISTORY OF THE SAVE COMMAND BELOW.

WHEN ZYMON WAS WRITTEN (IN 1981), ONLY THE KEMITRON TAPE INTERFACE, TPA-1 WAS AVAILABLE. THIS WAS DESIGNED FOR 300 BAUD OPERATION, AND SUFFERED FROM THE TENDENCY TO DRIFT AND CAUSE TAPE ERRORS UNLESS ITS MASTER CLOCK, TYPE 555, WAS REGULARLY READJUSTED. NOW (1984) MOST USERS HAVE THE INTERAK DTI-1 TAPE INTERFACE, WHICH PERMITS BAUD RATES UP TO 2400, HAS A QUARTZ CRYSTAL AS ITS MASTER CLOCK, AND A PHASE LOCKED LOOP TO COUNTERACT TAPE SPEED VARIATIONS. IN CONSEQUENCE, THE MAJORITY OF USERS ENJOY VIRTUALLY ERROR-FREE PERFORMANCE AT HIGH BAUD RATES OF 1200 OR EVEN 2400.

AT THE LOCATIONS GIVEN AT THE FOOT OF THIS PAGE, DELIBERATE DELAYS WERE BUILT INTO THE ZYMON SAVE ROUTINES; THESE DELAYS WERE TO ALLOW TIME FOR ZYMON TO RESPOND TO AN ERROR DETECTED DURING A LOAD. ZYMON'S RESPONSE WAS TO SCROLL THE SCREEN TO GIVE THE USER A RECORD OF THE ERROR LOCATION (TO THE NEAREST 8 BYTES). WITHOUT THE DELAY THERE WAS THE DANGER THAT SUBSEQUENT DATA MIGHT BE MISSED DURING THE TIME THE SCREEN SCROLL WAS BEING EXECUTED. AT 300 BAUD THE AMOUNT OF TIME WASTED DURING THE DELAY IS INSIGNIFICANT; REMOVING IT HARDLY SPEEDS UP THE SAVE AT ALL, BUT AT 2400 BAUD THE SCROLL DELAY IS VERY SIGNIFICANT, AND INCREASES THE TIME FOR A SAVE ALMOST FIVEFOLD.

BECAUSE TAPE SAVING AND LOADING IS SO RELIABLE USING THE DTI-1 CARD, AND BECAUSE MOST USERS FAVOUR THE FASTER BAUD RATES, BEING WILLING TO TAKE THEIR CHANCES REGARDING RECOVERY AFTER AN ERROR (MOST PEOPLE PRUDENTLY MAKE SEVERAL BACKUP COPIES OF ANY DATA THEY CONSIDER IMPORTANT), THE SUPPLIERS OF ZYMON HAVE DECIDED TO DELETE THE SCROLL DELAY FROM EPROMS SUPPLIED AFTER 29TH FEBRUARY 1984. (FURTHER INFORMATION HAS BEEN OR WILL BE PUBLISHED IN THE "INTERAKTION" USER GROUP NEWSLETTER.)

IF FOR SOME SPECIAL REASON YOU WOULD LIKE THE DELAY RESTORED, THE BYTES TO CHANGE ARE GIVEN BELOW. (AS ZYMON RUNS IN RAM IN CURRENT SYSTEMS YOU CAN MAKE THE CHANGES WITH ZYMON'S OWN MODIFY COMMAND, BUT REMEMBER THEN NOT TO EXECUTE A COLD START BEFORE THE SAVE COMMAND HAS BEEN USED TO STORE THE DATA ON THE CASSETTE - SEE PAGE 5 OF THIS MANUAL FOR DETAILS OF THE COLD START.)

(ALTHOUGH IT DOESN'T HAVE MUCH TO DO WITH THE PRESENT DISCUSSION, HERE IS A CONVENIENT PLACE TO MAKE A NOTE THAT AS ZYMON WAS WRITTEN SO LONG AGO (BEFORE THE DTI-1 EISTED) IT DOES NOT TAKE ADVANTAGE OF THE TAPE MOTOR RELAYS AND DUAL OUTPUTS ON THE DTI-1 CARD. THEREFORE WHEN OPERATING WITH ZYMON USE EITHER OF THE TWO CASSETTE CONNECTIONS, AND MANUAL CONTROL OF THE MOTORS.)

ADDRESS	SAVE WITHOUT DELAY	SAVE WITH DELAY
03B6	000000	CDF204
047C	000000	CDF204
0776	34	33

(THE LAST ENTRIES REPRESENT ONLY THE VERSION NUMBERS, AND NEED ONLY BE CHANGED IF MAKING THE ABOVE MODIFICATIONS PERMANENT. THE VERSION NUMBERS SHOWN WERE THOSE CURRENT WHEN THIS NOTE WAS PREPARED.)

RETURN POINTS

TO RETURN TO ZYMON AT THE END OF A PROGRAM USE:-

ADDRESS 0000....WILL COLD START ZYMON AND CLEAR THE VDU.

ADDRESS 0066....WILL WARM START ZYMON, THE SCREEN WILL NOT BE CLEARED BUT SCROLLED BY ONE LINE. THEN LINES 22,23 AND 24 WILL BE CLEARED AND THE FOLLOWING MESSAGE WILL BE PRINTED:-

```
 ZYMON

>E ADDR
```

ADDR IS THE CURRENT GHOST PC VALUE.

ADDRESS 009D....THIS BEING THE MINIMUM DISTURBANCE RE-ENTRY POINT. ONLY THE PROMPT AND CURSOR WILL BE DISPLAYED. THE SCREEN IS NOT SCROLLED.

ZYMON OVERVIEW

TO AID IN STUDYING THE LISTING HERE IS A MAP OF ZYMON.

```
0000 COLD START
0038 TRAP RE-ENTRY POINT
0066 WARM START
008E INLINE 24
00E4 DO LINE 24
00FD ERROR HANDLER
0110 COMMAND FOUND:GET DETAILS FROM THE TABLE
0131 GET THE COMMAND PARAMETERS
01A6 TRANSFER TO THE COMMAND DRIVER
01AC MODIFY
01CD ZERO
01D0 REGISTERS
01DC FILL
01EF EXECUTE
0248 TABULATE
0276 JUMP
02A9 PORT
02C2 VERIFY
02ED COPY
0324 BURN
0378 SAVE
03DF LOAD
0477 SUBROUTINES, STRINGS AND PROCEEDURES.
07BE COMMAND TABLE
```

SUBROUTINES

FOR DETAILS PLEASE CONSULT THE LISTING.

ADDR	NAME	FUNCTION
0491	FETCH A	THE A REGISTER IS LOADED WITH THE DAV VALID TAPE DATA. IT IS ALSO PRINTED TO (HL) AND (HL+1) AS ASCII/HEX.
04B1	SAVE A	THE DATA IN A IS SAVED TO TAPE AS AN 8 BIT BINARY NUMBER. ALSO THE ASCII/HEX OF THE DATA IS PRINTED TO (HL) AND (HL+1).
04C9	DEC IX	THE IX REGISTER IS DECREMENTED BY 1 AND THE FLAGS ARE SET TO INDICATE THE RESULT.
04DA	BURN IT	THE EPROM PROGRAMMER IS CYCLED ONCE TO BURN THE DATA WHICH IT IS ASSUMED TO BE LATCHED ON ITS INPUTS.
04F2	DELAY B	A DELAY WILL OCCUR WHICH EQUALS THE CONTENTS OF THE B REG TIMES 1ms (USING A 4MHZ CLOCK).
0596	ASCHEX TO BIN	THE ASCII/HEX DATA FROM (HL) AND (HL+1) IS CONVERTED TO AN 8 BIT BINARY IN THE C REGISTER.
05B2	PRINT REGS	THE GHOST REGISTER SET IS PRINTED TO THE VDU SCREEN.
05E4	ZERO GHOSTS	THE GHOST REGISTERS ARE ZEROED.
05EF	PRINTSTRING 1	A STRING POINTED TO BY HL IS PRINTED STARTING AT ADDRESS F2C1. THE ACTION IS TERMINATED BY HEX FF AT THE END OF THE STRING.
05F2	PRINTSTRING 2	A STRING POINTED TO BY HL IS PRINTED STARTING AT AN ADDRESS HELD IN THE DE REGISTER PAIR. TERMINATED BY HEX FF AT THE END OF THE STRING.
0602	SCROLL 1-23	LINES 1-23 ARE SCROLLED. LINE 23 IS THEN CLEARED.
0618	BIN-ASCHEX	THE 8 BIT BINARY IN THE C REGISTER IS CONVERTED TO TWO ASCII/HEX DIGITS WHCH ARE PRINTED TO ADDRESS (HL) AND (HL+1).
0635	RESTORE TRAP	THE SCRATCHPAD SAVED TRAPPED OP-CODE IS RESTORED TO THE SCRATCHPAD SAVED ADDRESS.
064B	CHEX	THE ASCII/HEX CHARACTERS POINTED TO BY THE HL REGISTER PAIR ARE CHECKED FOR VALID ASCII HEX. THE NUMBER OF CHARACTERS EXPECTED IS HELD IN THE B REGISTER.

SUBROUTINES CONTINUED

PLEASE CONSULT THE LISTING BEFORE USING THESE ROUTINES.

ADDR	NAME	FUNCTION
067A	DISP DE (DE)1	PRINTED STARTING AT ADDRESS F2E3 WILL BE THE ASCII/HEX OF THE DE REGISTER PAIR, FOLLOWED BY A SPACE AND THEN THE ASCII/HEX OF THE DATA POINTED TO BY THE DE REGISTER PAIR.
067D	DISP DE (DE)2	PRINTED STARTING AT THE ADDRESS HELD IN THE HL REGISTER PAIR WILL BE THE ASCII/HEX OF THE DE REGISTER PAIR, FOLLOWING THIS A SPACE IS PRINTED AND THEN THE DATA POINTED TO BY THE DE REGISTER PAIR IS PRINTED.
0694	DISP DE	THE ASCII/HEX EQUIVALENT OF THE DE REGISTER PAIR IS PRINTED STARTING AT THE ADDRESS POINTED TO BY THE HL REGISTER PAIR.
06A4	E PC	PRINTED TO LINE 24 WILL BE "E ADDR" WHERE ADDR IS THE CURRENT CONTENTS OF THE GHOST PROGRAM COUNTER.
06BB	HL-DE	THE DE REGISTER PAIR IS SUBTRACTED FROM THE HL REGISTER PAIR. BOTH DE AND HL ARE RESTORED AND THE FLAGS SHOW THE RESULT: I.E.:-

	S	Z	C
HL=DE	0	1	0
HL>DE	0	0	0
HL<DE	1	0	1

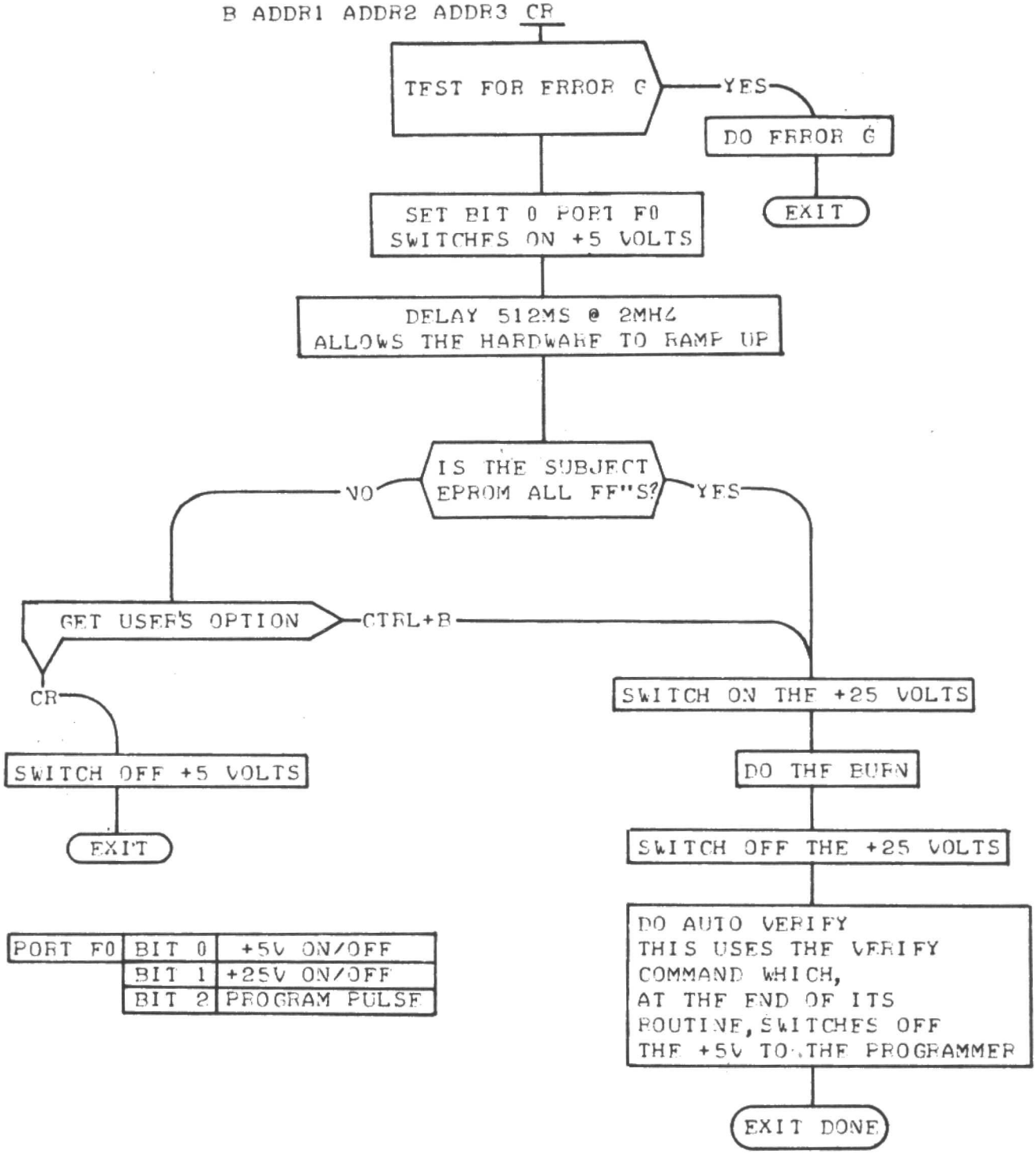
06C3	CVDU	THE VDU SCREEN IS CLEARED.
06D7	CLINEND	HEX 20 IS WRITTEN TO THE ADDRESS POINTED TO BY THE HL REGISTER PAIR, 1 IS ADDED TO HL AND THE PROCESS IS REPEATED UNTIL THE LOW 5 BITS OF THE L REGISTER ARE ALL ZERO, I.E. LINE END ON THE VDU.
06E2	KBR	THE KEYBOARD PORT, (40) IS READ. IF THERE IS NO STROBE, I.E. NO KEY, THE A REGISTER IS SET TO ZERO AND THE Z FLAG IS SET TO ONE. IF THERE IS A STROBE, THE VALID KEY IS RETURNED IN THE A REGISTER AFTER THE STROBE IS OVER. THE ZERO FLAG=0 FOR A VALID KEY.
06F0	MV24-23	THE CONTENTS OF LINE 24 ON THE VDU ARE MOVED TO LINE 23 ON THE VDU.

EPROM POWER SEQUENCE FLOWCHART

ZYMON SOFTWARE FOLLOWS THIS ROUTINE TO RAMP UP THE PROGRAMMER POWER SUPPLIES. IT USES PORT F0 BITS 0, 1 AND 2.

AT THE COMMAND

B ADDR1 ADDR2 ADDR3 CR



PORT F0	BIT 0	+5V ON/OFF
	BIT 1	+25V ON/OFF
	BIT 2	PROGRAM PULSE

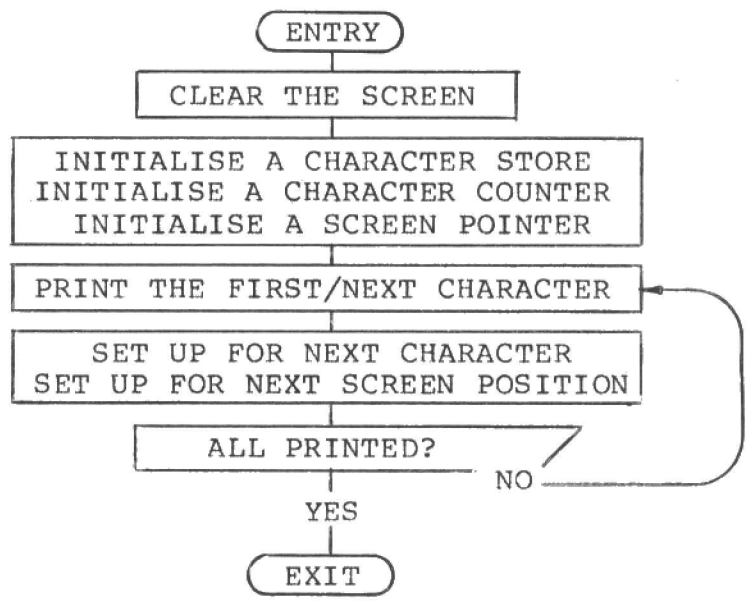
A PROGRAM BUILT WITH ZYMON 2

THE MOST EFFECTIVE WAY TO LEARN HOW TO USE ZYMON 2 IS TO WRITE A PROGRAM. AS THERE ARE SEVERAL WAYS TO DO THIS I SHALL DISCUSS MY OWN METHOD. THE FIRST STEP IS TO CHOOSE A PROGRAMMING TASK, THEN YOU SHOULD ASK AND ANSWER AS MANY QUESTIONS AS POSSIBLE ABOUT THE TASK, RECORDING THE ANSWERS FOR LATER USE. I.E.:-

1) WHAT SHOULD THE PROGRAM DO?.....DISPLAY THE FIRST FEW CHARACTERS ON THE VDU.
2) HOW MANY CHARACTERS?.....64 FOR NOW.
3) WHICH ONE FIRST?.....THE ONE WHOSE CODE IS 00
4) TO WHICH LINE OF THE VDU?.....LINE 6 (ADDRESS F0A0).

YOU SHOULD CONTINUE THIS PROCESS UNTIL YOU CANNOT THINK OF ANY MORE QUESTION/ANSWER SETS. THIS THEN GIVES A PROGRAM SPECIFICATION FROM WHICH THE ACTUAL PROCESS CAN BE CREATED.

NEXT WE NEED TO PRODUCE A DESCRIPTION OF THE PROCESS AND THIS IS BEST DONE BY A FLOW CHART. I.E.:-



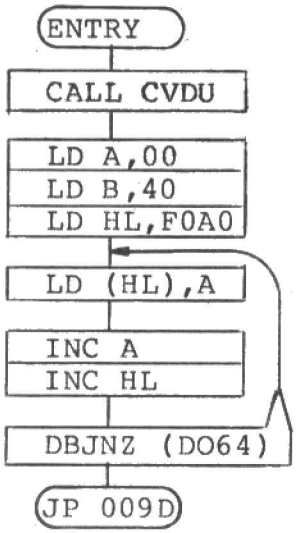
THIS THEN IS AN OVERVIEW. NOW SOME DEFINITIONS ARE NEEDED.

LET THE A REGISTER BE THE CHARACTER CODE STORE
LET THE B REGISTER BE THE CHARACTER COUNTER STORE
LET THE HL REGISTER PAIR HOLD THE SCREEN ADDRESS.

NOTICE THAT CHOOSING THE B REGISTER FOR THE CHARACTER COUNTER WILL ALLOW US TO USE THE DBJNZ INSTRUCTION TO OUR ADVANTAGE. THIS WILL BECOME CLEAR LATER IN THIS DISCUSSION.

WE NOW HAVE A PROGRAM SPECIFICATION AND SOME REGISTER DEFINITIONS, AND SO WE CAN PROCEED TO THE NEXT PHASE WHICH IS TO RE-WRITE OUR FLOW CHART USING Z80 ASSEMBLY LANGUAGE INSTRUCTIONS.

THIS IS OUR PROGRAM CONVERTED TO Z80 ASSEMBLY LANGUAGE:-



GIVE IT A NAME, SAY "PRINCHARS".

USING THE EXISTING ZYMON ROUTINE.

INITIALISE THE CHARACTER STORE.
INITIALISE THE CHARACTER COUNTER.
INITIALISE THE SCREEN POINTER.

(DO64) "DO 64" PRINT THE FIRST/NEXT CHARACTER.

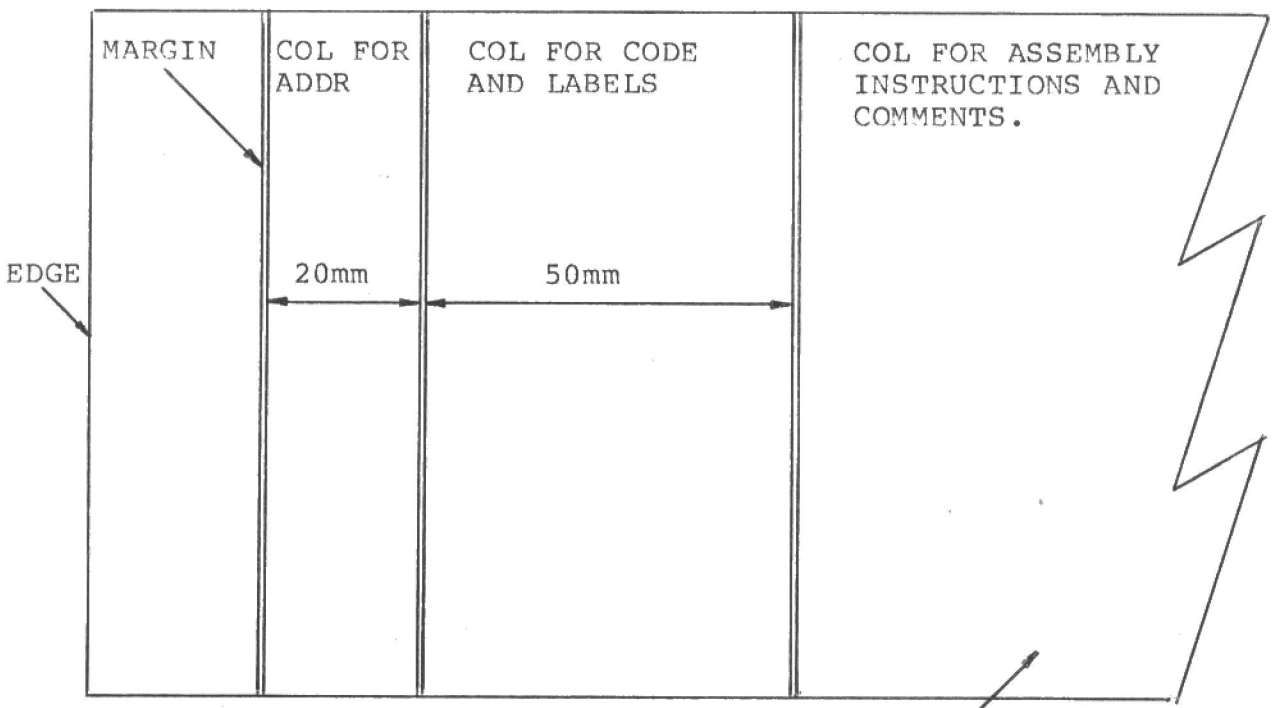
SET UP THE NEXT CHARACTER.
SET UP THE NEXT SCREEN POSITION.

IF NOT DONE JUMP TO (DO64)

EXIT TO ZYMON. MIN DISTURBANCE RE-ENTRY POINT.

NOTICE WE WILL RETURN TO ZYMON AT THE END OF THE PROCESS, WITHOUT DISTURBING THE SCREEN CONTENTS. AT THIS STAGE OF THE DEVELOPMENT YOU CAN ALTER THINGS VERY EASILY, SO SPEND QUITE A LOT OF TIME HERE. WHEN YOU ARE CONTENT MOVE ON TO THE NEXT STEP, WHICH IS THE CODING PROCESS.

TO FORMAT THE CODING, USE MARGINATED NARROW FEINT A4 PAPER, HAVING DRAWN TWO VERTICAL LINES DOWN IT AS:-



USING THIS SHEET WRITE, IN THE RIGHT HAND COLUMN, THE ASSEMBLY LANGUAGE INSTRUCTIONS PREVIOUSLY DEFINED.

YOUR CODING SHEET SHOULD LOOK LIKE THIS:-

```
|      |      |      |  CALL ⌐ ⌐      CLEAR THE VDU
|      |      |      |  LD A,00       FIRST CHAR CODE=00
|      |      |      |  LD B,40       NUMBER OF CHARS=64 DEC
|      |      |      |  LD HL,F0A0    FIRST SCREEN POS=LINE 6
|      |      | DO64 |  LD (HL),A     PRINT FIRST/NEXT CHAR
|      |      |      |  INC A         SET NEXT CHAR CODE
|      |      |      |  INC HL        SET NEXT VDU POSITION
|      |      |      |  DBJNZ(DO64)   IF NOT DONE JUMP TO (DO64)
|      |      |      |  JP 009D       EXIT DONE
```

NOW FILL IN ALL THE HEX OP-CODES THAT YOU ARE ABLE TO. WHEN YOU DON'T KNOW THE CODE DRAW .. TO INDICATE THE REQUIREMENT TO HAVE A BYTE THERE FOR LATER INSERTION. THIS WILL GIVE YOU:-

```
|      |  CD ⌐ ⌐        |      |  CALL ⌐ ⌐      CLEAR THE VDU
|      |  3E 00         |      |  LD A,00       FIRST CHAR CODE=00
|      |  06 40         |      |  LD B,40       NUMBER OF CHARS=64 DEC
|      |  21 A0 F0      |      |  LD HL,F0A0    FIRST SCREEN POS=LINE 6
|      |  77            | DO64 |  LD (HL),A     PRINT FIRST/NEXT CHAR
|      |  3C            |      |  INC A         SET NEXT CHAR CODE
|      |  23            |      |  INC HL        SET NEXT VDU POSITION
|      |  10 ⌐          |      |  DBJNZ(DO64)   IF NOT DONE JUMP TO (DO64)
|      |  C3 9D 00      |      |  JP 009D       EXIT DONE
```

WE DON'T KNOW TWO THINGS AT THIS POINT, THESE BEING THE ADDRESS OF THE CLEAR SCREEN SUBROUTINE AND THE JUMP DISPLACEMENT FROM THE DBJNZ INSTRUCTION TO THE LABEL DO64, BUT WE DO KNOW HOW MANY BYTES ARE REQUIRED AND SO WE CAN FIND THESE OUT LATER. WE NOW HAVE SUFFICIENT INFORMATION TO CHOOSE A STORE LOCATION FOR OUR PROGRAM. SO CHOOSING TO START AT 0800 (SINCE MOST PEOPLE WILL HAVE RAM THERE) MEANS WE CAN WRITE IN THE ADDRESSES. THE CODING SHEET WILL LOOK LIKE THIS:-

```
|0800 |  CD ⌐ ⌐        |      |  CALL ⌐ ⌐      CLEAR THE VDU
|0803 |  3E 00         |      |  LD A,00       FIRST CHAR CODE=00
|0805 |  06 40         |      |  LD B,40       NUMBER OF CHARS=64 DEC
|0807 |  21 A0 F0      |      |  LD HL,F0A0    FIRST SCREEN POS=LINE 6
|080A |  77            | DO64 |  LD (HL),A     PRINT FIRST/NEXT CHAR
|080B |  3C            |      |  INC A         SET NEXT CHAR CODE
|080C |  23            |      |  INC HL        SET NEXT VDU POSITION
|080D |  10 ⌐          |      |  DBJNZ(DO64)   IF NOT DONE JUMP TO (DO64)
|080F |  C3 9D 00      |      |  JP 009D       EXIT DONE
|0812 |                |      |
```

THE LAST ADDRESS IS PUT IN TO INDICATE THE NEXT FREE STORE LOCATION, FOR THE NEXT ROUTINE, SHOULD THERE EVER BE ONE. THAT'S AS FAR AS WE CAN GO WITHOUT OBTAINING THE UNKNOWN BYTES OF THE SUBROUTINE AND THE JUMP DISPLACEMENT. SO THAT'S NEXT.

TO OBTAIN THE SUBROUTINE ADDRESS LOOK IN THIS MANUAL AT PAGE 35 WHERE YOU WILL FIND CVDU CALLED AT 06C3. TO GET THE JUMP DISPLACEMENT TYPE TO ZYMON:-

```
J 080D 080A CR      THIS WILL RETURN WITH:-
J 080D 080A FB      THE FB IS OUR REQUIRED BYTE.
```

NOW WE HAVE ENOUGH INFORMATION TO COMPLETE THE CODING:-

```
0800 | CD C3 06            |  CALL 06C3.   CLEAR THE VDU
0803 | 3E 00               |  LD A,00      FIRST CHAR CODE=00
0805 | 06 40               |  LD B,40      NUMBER OF CHARS=64 DEC
0807 | 21 A0 F0            |  LD HL,F0A0   FIRST SCREEN POS=LINE 6
080A | 77           DO64   |  LD (HL),A    PRINT FIRST/NEXT CHAR
080B | 3C                  |  INC A        SET NEXT CHAR CODE
080C | 23                  |  INC HL       SET NEXT VDU POSITION
080D | 10 FB               |  DBJNZ(DO64)  IF NOT DONE JUMP TO (DO64)
080F | C3 9D 00            |  JP 009D      EXIT DONE
0812 |                     |
```

THIS THEN IS OUR COMPLETED PROGRAM. NOW TO TRY IT OUT.

FIRST OF ALL, TO GIVE US SOME PROTECTION, FILL THE STORE AREA WITH HEX FF. (ARTIFICIAL TRAPS):-

```
F 0800 0F00 FF CR
```

NOW IF WE JUMP OUT OF OUR STORE AREA ACCIDENTALLY WE SHOULD HIT ONE OF THESE FF CODES AND GET AN ARTIFICIAL TRAP REPORTED.

THEN ENTER THE PROGRAM USING THE MODIFY COMMAND:-

```
M 0800 CD CR
M 0801 C3 CR
M 0802 06 CR
   --
   --
   --
M 0811 CD CR....AND IT'S ALL IN.
```

NEXT, TO CHECK IT OUT TYPE:-

```
T 0800 CR.......THIS WILL TABULATE THE 64 (DECIMAL) BYTES
                FROM 0800 AND YOU CAN COMPARE THEM AGAINST
                YOUR LISTING.
```

FINALLY SAVE THE PROGRAM TO TAPE WITH:-

```
S 0800 0811 CR..HAVING STARTED THE TAPE OF COURSE.
```

NOW WE ARE IN A POSITION TO TRY THE PROGRAM. IF IT CRASHES IT CAN BE RELOADED FROM TAPE, IF IT JUMPS INCORRECTLY IT SHOULD HIT AN ARTIFICIAL TRAP. YOU COULD IF YOU WISHED RUN THIS PROGRAM NOW. BUT BETTER WOULD BE TO USE THE TRAP MECHANISM TO ENSURE THE CORRECT ACTIONS TAKE PLACE. SEE NEXT PAGE FOR DETAILS.

TO TAKE THE PROGRAM THROUGH ITS ACTIONS TYPE:-

```
>E 0800 080A CR
```

THE VDU WILL NOW DISPLAY:-

```
 PC=080A SP=0FC5 IX=0000 IY=0000
 AF=0000 BC=4000 DE=0000 HL=F0A0

>E 080A
```

THIS SHOWS THAT THE REGISTERS HAVE BEEN CORRECTLY LOADED. NOW TYPE ONTO THE END OF THE " E 080A", (ON LINE 24), 080D CR.....AND THE DISPLAY SHOWS:-

```
 PC=080D SP=0FC5 IX=0000 IY=0000
 AF=0100 BC=4000 DE=0000 HL=F0A1

>E 080D
```

THE FIRST CHARACTER HAS BEEN PRINTED NEAR THE TOP OF THE VDU. UNFORTUNATELY WITH MANY VDU CHARACTER GENERATORS THIS IS A BLANK, SO IT IS A LITTLE HARD TO SEE. OBSERVE THAT THE A REGISTER HAS BEEN INCREMENTED BY ONE, AND THE HL REGISTER PAIR NOW POINTS TO THE NEXT FREE SCREEN POSITION. NOW AFTER THE " E 080D", (ON LINE 24), TYPE 080C CR.....AND THE DISPLAY SHOWS:-

```
 PC=080C SP=0FC5 IX=0000 IY=0000
 AF=0200 BC=3F00 DE=0000 HL=F0A1

>E 080C
```

NOTICE THAT THE B REGISTER HAS BEEN COUNTED DOWN BY ONE. WE HAVE JUST BEEN AROUND THE LOOP ONCE. TO GO ROUND AGAIN TYPE TO LINE 24 THE NEXT TRAPS. (I.E. YOU JUST ADD THE TRAP ADDRESS AND CR.)

```
>E 080C 080D CR....TO MOVE THE TRAP TO THE JUMP. THEN TYPE
>E 080D 080C CR....AND YOU WILL HAVE GONE ROUND THE LOOP AGAIN.
```

WHEN YOU'VE BEEN ROUND A FEW TIMES, AND WATCHED THE CHARACTERS APPEARING, TYPE CTRL-C AND:-

```
>E 0800 CR
```

THE PROGRAM SHOULD RUN!!!!!

THAT'S THE END OF THIS SHORT DISCUSSION ON PROGRAMMING. I HOPE YOU ENJOY USING ZYMON, AND I LOOK FORWARD TO RUNNING ONE OF YOUR PROGRAMS, ON MY SYSTEM, VERY SOON.

WISHING YOU THE BEST OF LUCK.

Bob Eldridge.

COMMAND SUMMARY (ALL FOLLOWED BY CR)

B ADDR1 ADDR2 ADDR3	BURNS THE EPROM STARTING AT ADDR3 WITH THE DATA FROM ADDR1 TO ADDR2.
C ADDR1 ADDR2 ADDR3	COPIES THE DATA BLOCK FROM ADDR1 TO ADDR2 INCLUSIVE, TO THE DESTINATION BLOCK STARTING AT ADDR3.
E	SETS UP AN EXECUTE COMMAND IN THE FORM "E GHOST PROGRAM COUNTER".
E ADDR1	LOADS THE Z80 WITH THE GHOST REGISTERS, AND EXECUTES CODE FROM ADDR1.
E ADDR1 ADDR2	SETS A TRAP AT ADDR2, HAVING SAVED THE CONTENTS. THEN LOADS THE Z80 WITH THE GHOST REGISTERS AND EXECUTES CODE FROM ADDR1.
F ADDR1 ADDR2 VAL	FILLS THE MEMORY FROM ADDR1 TO ADDR2 WITH THE VAL.
J ADDR1 ADDR2	RETURNS THE REQUIRED DISPLACEMENT TO JUMP RELATIVE FROM ADDR1 TO ADDR2.
L	LOADS THE NEXT SEQUENTIAL FILE FROM TAPE TO STORE, USING THE ON TAPE ADDRESS.
L ADDR1	LOADS THE NEXT SEQUENTIAL FILE FROM TAPE TO STORE STARTING AT ADDR1.
M ADDR1	READS AND DISPLAYS THE CONTENTS OF ADDR1.
M ADDR1 VAL	WRITES VAL TO ADDR1 AND READS IT BACK.
P PT	DISPLAYS THE CONTENTS OF PORT PT.
P PT VAL	WRITES VAL TO PORT PT. NOT READ BACK.
R	DISPLAYS THE GHOST REGISTER SET.
S ADDR1 ADDR2	SAVES ON TAPE THE FILE FROM ADDR1-ADDR2.
T ADDR1	TABULATES ON THE VDU 64 (DECIMAL) LOCATIONS.
V ADDR1 ADDR2 ADDR3	COMPARES THE BOCK ADDR1-ADDR2 WITH THE BLOCK STARTING AT ADDR3.
Z	ZEROS THE GHOST REGISTER SET.

KEY COMMANDS:-	BS	(CTRL-H)	BACKSPACE (CURSOR LEFT, DESTRUCTIVE)
	FF	(CTRL-L)	CLEAR SCREEN.
	EXT	(CTRL-C)	INPUT COMMAND.
	CR	(CTRL-M)	OBEY THE LINE 24 COMMAND.
	[		CURSOR LEFT (NON-DESTRUCTIVE)
	]		CURSOR RIGHT (NON-DESTRUCTIVE)

ERROR CODES

B.....BLANK (SPACE) EXPECTED IN THIS POSITION.
C.....CASSETTE ERROR, LOAD TERMINATED. THE CHECKSUM OF THE BYTE COUNT AND START ADDRESS DID NOT COMPARE. LINE 24 DISPLAYS BBCC SSAA CS.
F.....TOO FEW PARAMETERS GIVEN.
G.....SOURCE START IS GREATER THAN SOURCE END.
H.....THE INDICATED VALUE IS NOT ASCII/HEX.
J.....THE RELATIVE JUMP IS OUT OF RANGE.
L.....THE PARAMETER IS TOO LONG.
M.....TOO MANY PARAMETERS GIVEN.
P.....THE PARAMETER IS NOT IN THE EXPECTED PLACE.
S.....THE PARAMETER IS TOO SHORT.
T.....ATTEMPT TO SET A TRAP IN A NON RAM LOCATION.
U.....UNEXPECTED TRAP DETECTED.
X.....COMMAND LETTER UNKNOWN.

GHOST REGISTER MAP.

0FC6 0FC7	C P	PC
0FC8 0FC9	P S	SP
0FCA 0FCB	X I	IX
0FCC 0FCD	Y I	IY
0FCE 0FCF	F A	AF
0FD0 0FD1	C B	BC
0FD2 0FD3	E D	DE
0FD4 0FD5	L H	HL

ZYMON'S SCRATCHPAD

0FD6 0FD7	COMMAND ADDRESS
0FD8	ALWAYS ZERO
0FD9	LENGTH 3
0FDA	LENGTH 2
0FDB	LENGTH 1
0FDC	EXPECTED
0FDD	FOUND
0FDE 0FDF	PARAMETER 1
0FE0 0FE1	PARAMETER 2
0FE2 0FE3	PARAMETER 3
0FE4 0FE5	TRAP ADDRESS
0FE6	TRAP OP-CODE

(ZYMON'S OWN STACK IS FROM 0FE8 TO 0FFF INCLUSIVE.)